1990

Fuzzy logic control of a fluidized bed combustor

Stephen J. Koffman
Iowa State University

Fuzzy logic control of a fluidized bed combustor

by

Stephen J. Koffman

A Thesis Submitted to the

Graduate Faculty in Partial Fulfillment of the

Requirements for the Degree of

MASTER OF SCIENCE

Major: Mechanical Engineering

Signatures have been redacted for privacy

Iowa State University
Ames, Iowa
1990

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

I gratefully acknowledge the National Science Foundation's financial support for this research through their "Creativity Awards for Undergraduate Engineering Students" program. I sincerely thank Dr. Robert C. Brown and Dr. R. Rees Fullmer for their valuable assistance throughout the course of this work.

NOMENCLATURE

$\mu_{R_n}(t_i, f_j)$	=	grade of membership over temperature/fuel space for $n^{\underline{th}}$ rule of the hypothetical system
$\mu_R(t_i, f_j)$	=	grade of membership for overall fuzzy relation between temperature and fuel
$\mu_{T_{low}}(t_i)$	=	grade of membership for t_i element of "low" temperature set
$\mu_{F_{high}}(f_j)$	=	grade of membership for f_j element of "high" fuel feed rate set
$\mu_T(t_i)$	=	grade of membership for fuzzy temperature input set to compositional rule
$\mu_F(f_j)$	=	grade of membership for fuzzy fuel feed rate output set of compositional rule
$\bar{f}_{MOM}$	=	control action produced by mean-of-maximum defuzzification technique
$\bar{f}_{COA}$	=	control action produced by center-of-area defuzzification technique
f_k's	=	fuel feed rate elements having maximum membership for a particular output fuel feed rate set
N	=	number of elements with maximum membership
n	=	number of elements in a fuzzy set

1. INTRODUCTION

Increased concerns over acid rain, global warming, and depletion of inexpensive fuel supplies have led researchers to seek cleaner and more efficient combustion techniques. Recently, fluidized bed combustors (FBCs) have attracted interest for generating industrial power and steam because of their ability to burn low-grade fuels while maintaining strict emission standards. FBCs have several advantages over conventional boilers. They permit burning of some substances not previously thought as practical fuels such as biomass, solid and slurry wastes, and high-sulfur fossil fuels [1]. FBCs also allow low-level pollutant emissions to be easily maintained. Because NO_x production increases with temperature and coal-fired FBCs operate near 1600°F as opposed to 3000°F for conventional boilers [1], FBCs produce smaller amounts of nitrogen oxides than conventional boilers [2]. Also, sorbent material such as limestone can be added to FBCs to absorb sulfur dioxide as it is formed during the combustion process [2]. These advantages make FBCs attractive for meeting future power needs; however, these combustors are not without disadvantages.

FBCs have coupled dynamics and severe nonlinearities that have frustrated attempts to effectively automate their operation using conventional control techniques [3], especially during start-up. In the past, combustor operation has depended on frequent human intervention. For coal-fired FBCs to be an economical energy system for small industries, their operation must have a high degree of automation and reliability — the expense of human labor devoted to combustor operation and maintenance may make FBCs prohibitive. Furthermore, because small industries tend to use power intermittently, FBCs in such industries are expected to have frequent restarts after periods of non-use. Therefore, efficient start-up operation will be required of these combustors. Conventional control algorithms may not be suitable for this task since parameters of the process, upon which classical design methods are based, rapidly change during combustor transients. To adequately automate these combustors for start-up and steady-state operation, new approaches need to be examined.

Fuzzy logic — an artificial intelligence technique — can be employed to exploit the wealth of information human experts have learned about complex systems while attempting to control them. This information is usually of a qualitative nature that is unusable by rigid conventional control techniques. Fuzzy logic, used as a control method, manipulates linguistically-expressed, heuristic knowledge from a human expert to derive control actions for a described system. As an alternative approach to classical controls, this thesis examines fuzzy logic for start-up control and steady-state regulation of a two-bed fluidized bed combustor. Preliminary fuzzy controller design is facilitated by simulating the controller on a computer model of the FBC. Further controller tuning and redesign are based upon actual test runs. To validate the fuzzy logic approach, the fuzzy controller is compared to a classical PI controller designed by the Ziegler-Nichols transient-response method [4].

2. FUZZY LOGIC THEORY

2.1 Overview

Fuzzy logic, developed by L. A. Zadeh [5, 6] in the mid 1960s, is an artificial intelligence technique that manipulates expert human knowledge to make decisions in ill-defined systems. Fuzzy logic systems are a subset of expert systems using approximate reasoning techniques based on qualitative, imprecise information. The idea in many expert systems is to extract precise expert knowledge from a person who is often unable to exactly quantify his actions. Fuzzy logic was developed to utilize this valuable (though inexact) information that would normally be difficult or impossible to exploit otherwise.

In the early 1970s E. H. Mamdani and his cohorts [7, 8, 9] explored using this technique to control complex dynamic systems previously controlled by skilled human operators. Since this time, many researchers [10, 11, 12, 13, 14] have applied fuzzy logic with good success to a wide variety of complex systems.

Fuzzy logic, being compatible with human thought, is comprised of two main components — an information base and an inference engine. The information base is a set of linguistic rules which qualitatively describe the system or actions to be performed on the system. These rules are gleaned from an expert. Because humans communicate and think in imprecise terms, the rules are expressed in qualitative (fuzzy) terms. Numerical meaning is given to the qualitative terms of the rules via fuzzy sets. The rules and fuzzy sets are operated upon by an inference engine to infer actions or decisions. As in the case of process control, the rules give qualitative control actions for qualitative operating points of a system whereas the inference mechanism gives quantitative actions for quantitative operating points.

2.2 Rules

The first step in implementing a fuzzy controller is to quiz the human operator about his actions for varying system states. Because of the imprecision with which humans think, the operator is often unable to precisely describe his actions to precisely asked questions about the system. Therefore, most questions should be asked in approximate terms. Responses to questions will most likely be in imprecise terms[1]. These terms will be descriptive of the system, e.g., high, hot, medium, not much, nearly zero, etc. Also, since the operator may have difficulty discriminating between control actions for two states having small differences, questions should cover moderate-to-large differences in system states. The fuzzy inference mechanism will derive control actions for those states not listed in the rules.

Knowledge extracted from the expert is stated in rule form. In general, rules have the following structure:

If $X=\{\text{antecedents}\}$ then $Y=\{\text{consequents}\}$.

[1]Depending upon the operator and control goals, it may be possible to obtain a definite response to a precisely asked question.

where X and Y are vectors. This type of arrangement corresponds to a multiple-input/multiple-output (MIMO) system. For single-input/single-output (SISO) systems, X and Y reduce to scalars. The rules are interpreted as: If the antecedents of the rule are met, then apply the consequents to the system. In the case of process control, X represents process outputs of the system (also controller inputs); Y represents process inputs (also controller outputs) that are manipulated to affect changes in process outputs. Antecedents and consequents, being expressed linguistically, are state descriptions of system outputs and inputs, respectively.

Consider a hypothetical SISO system which has temperature as the system state (output) and fuel feed rate as the manipulated variable (input). A rule appropriate to this system might be:

If temperature is "low", then fuel feed rate is set to "high".

As temperature of the system approaches "low", fuel feed rate is increased to "high". "Low" and "high" don't have any numerical meaning to this point; however, they give good qualitative information on how to control the system.

A complete set of rules for this system would specify all changes to fuel feed rate for all possible qualitative values ("fuzzy states") of temperature. Hence, if temperature is described by the three fuzzy states of "low", "ok", or "high", the number of rules is equal to three — one rule for each fuzzy state of temperature. The complete rule set for this hypothetical system follows:

1. If temperature is "low", then fuel feed rate is set to "high".

2. If temperature is "ok", then fuel feed rate is set to "normal".

3. If temperature is "high", then fuel feed rate is set to "low".

After obtaining a suitable set of rules for a particular system, the qualitative terms for both the antecedents and consequents of each rule must be defined using fuzzy sets as described below.

2.3 Fuzzy Sets

A normal set is a collection of items having a common, definite feature. For example, consider a set of temperatures above 1600°F. Either a particular temperature belongs to this set, or it does not. Likewise, a fuzzy set is a collection of items also having a common feature, but the elements of this set have this feature to varying degrees. Each element of a fuzzy set has graded membership ranging between 0.0 (no membership) and 1.0 (full membership) to the set. "Fuzzy" refers to the fact that these sets do not have sharp boundaries between non-members and members. For example, consider the hypothetical system and "ok" temperature fuzzy set from the previous section. The control objective for this system is to achieve and maintain 1600°F. Since it is the target temperature, 1600°F is 100% "ok", i.e., it has 1.0 membership to the "ok" temperature set. Temperatures near 1600°F are certainly more desirable than those further away. Hence, the closer a temperature is to 1600°F, the larger its membership is to "ok".

A relation giving grades of membership for each element of a fuzzy set is known as a membership function. Each fuzzy set is characterized by its membership function. Figure 2.1 is a plot of the membership function for "ok" temperature. Notice that temperatures below 1400°F and above 1800°F are so far away from 1600°F that

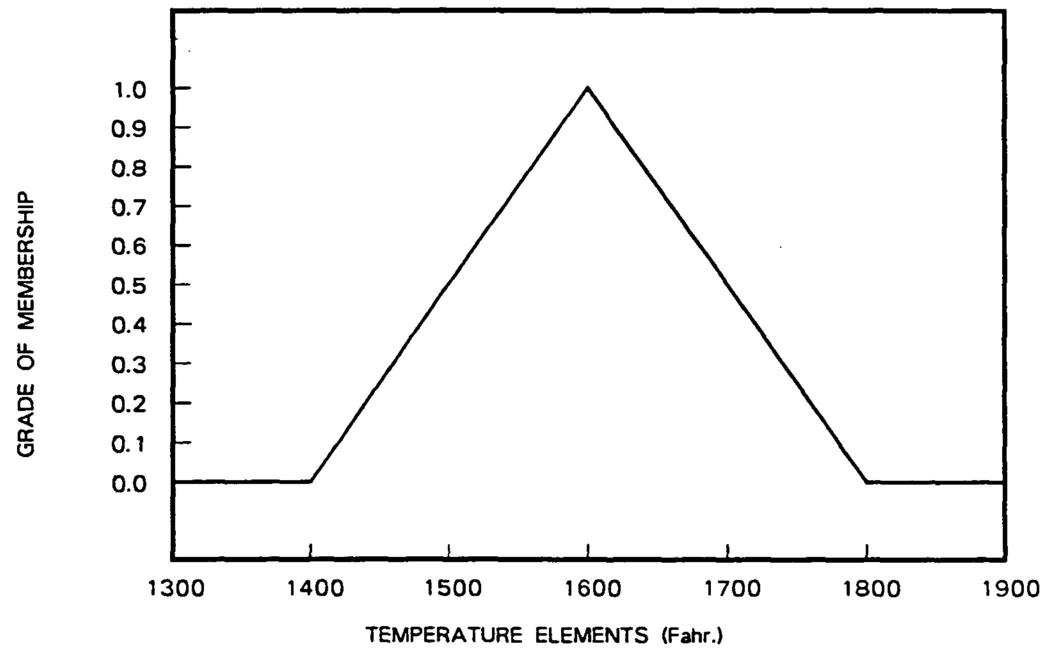

Figure 2.1: Membership Function for "Ok" Temperature of Hypothetical System

their memberships to the "ok" fuzzy set are 0.0. This membership function was subjectively defined by this author for the hypothetical system. Another person may have defined it differently. Therefore, it is imperative that the human expert who is being quizzed determine the membership function associated with each fuzzy term. Although membership functions may have curved shapes, most functions will probably be triangular because humans tend to think linearly when comparing quantities or magnitudes.

All fuzzy sets describing an item are defined on a specific space known as the *universe of discourse* for that item. Again using the hypothetical system as an example, "low", "ok", and "high" temperatures are defined on the temperature universe of discourse; fuzzy sets for fuel feed rate are defined on the fuel feed rate universe of discourse.

As seen, membership functions give numerical identity, though somewhat vague, to antecedents and consequents of the rule set. Rules with corresponding fuzzy sets comprise the information base. The following section describes how rules and fuzzy sets are manipulated by an inference engine to produce control actions or decisions.

2.4 Inference Mechanism

2.4.1 Purpose

The purpose of the inference mechanism is two-fold. First, it gives values of the consequents when the antecedents are not at 100% of their value. Secondly, the mechanism acts as an interpolator between neighboring rules. At any given instant, more than one rule governs the action of the controller. The inference mechanism manages both of these coupled problems simultaneously. An example of each is given below.

Recall the previous scenario with temperature as input and fuel feed rate as output of a system. Let all fuzzy sets for this system be defined as in Figure 2.2. Consider the rule: If temperature is "ok", then fuel feed rate is set to "normal". Intuitively, if temperature is 100% "ok", i.e., measured temperature is 1600°F, then fuel feed rate should be set to 100% "normal" (a value of 5 lb/hr, see Figure 2.2) as the rule states. But consider the case in which temperature is near 1550°F. Fuel feed rate for this temperature is not obvious, though it clearly is greater than 100% "normal". This is the first area where the inference mechanism plays an important role.

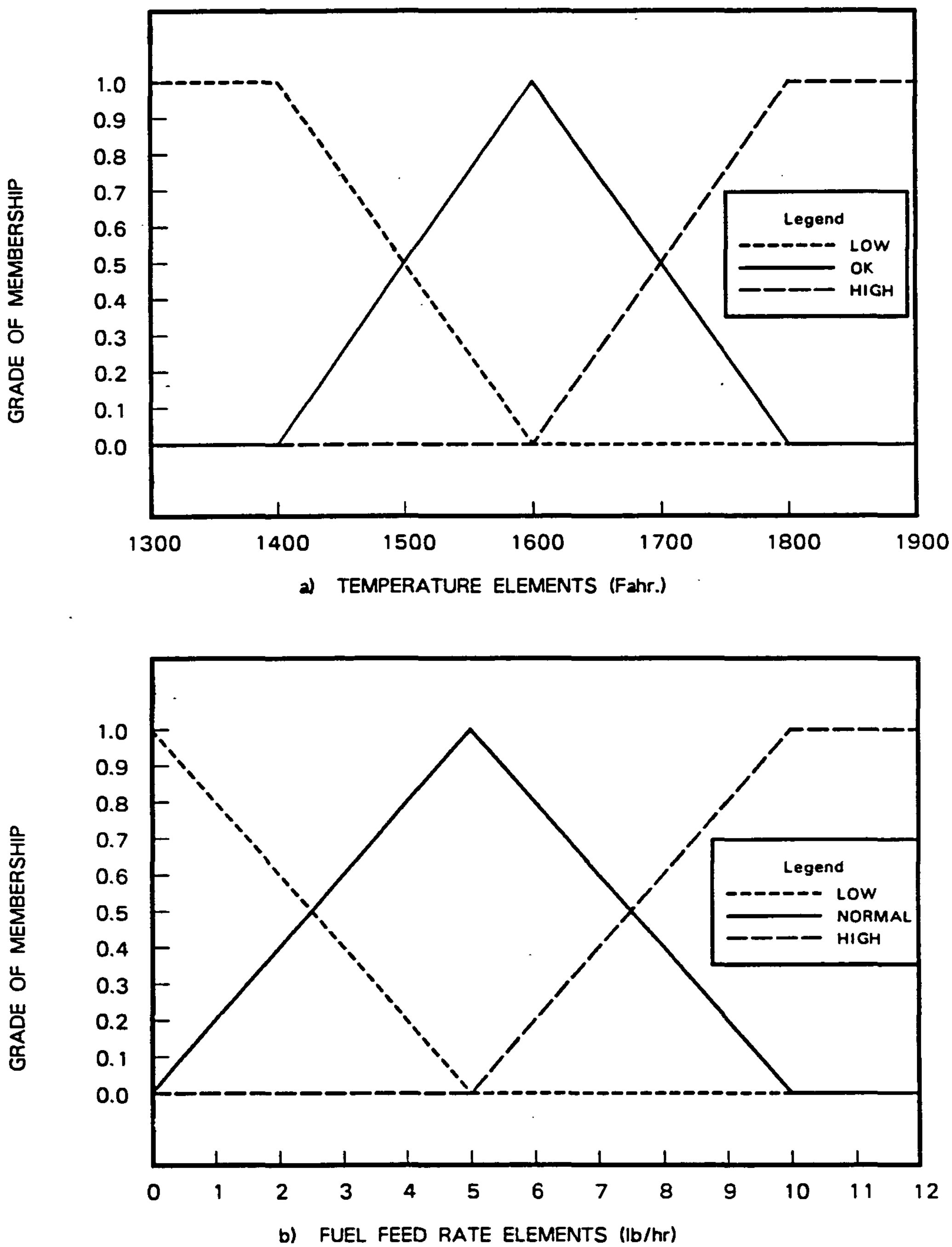

Figure 2.2: Membership Functions of Hypothetical System for: a) "Low", "Ok", and "High" Temperature, and b) "Low", "Normal", and "High" Fuel Feed Rate

Now consider the rule: If temperature is "low", then fuel feed rate is set to "high". Let measured temperature be 1550°F again. Not only does this rule affect the outcome, but the previous rule does as well because 1550°F is an element (with non-zero membership) in both the "ok" and "low" fuzzy sets of temperature. This type of interaction occurs whenever fuzzy sets for antecedents of different rules have overlapping membership functions. This is the second area where the inference mechanism becomes important.

2.4.2 Relation matrix

Each rule of the rule set forms a fuzzy relation that can be represented mathematically on an Euclidean space as the intersection of the membership functions of the antecedents and the consequents. This intersection is formed by a minimum operation between membership functions. Each controller input and output has its own axis — universe of discourse — upon which the fuzzy sets describing the inputs and outputs are respectively defined. For SISO systems, having one universe of discourse for the input and one for the output, this intersection can be represented in 3-D Cartesian space.

Figure 2.3 shows the intersection (Cartesian product) for the first rule of the hypothetical SISO system: If temperature is "low", then fuel feed rate is set to "high". Temperature forms one of the horizontal axes (universe of discourse), and fuel feed rate forms the other. The vertical axis has the grade of membership for this fuzzy relation. The surface in Figure 2.3 was generated from the equation:

$$\mu_{R_1}(t_i, f_j) = \min[\mu_{T_{low}}(t_i), \mu_{F_{high}}(f_j)]. \tag{2.1}$$

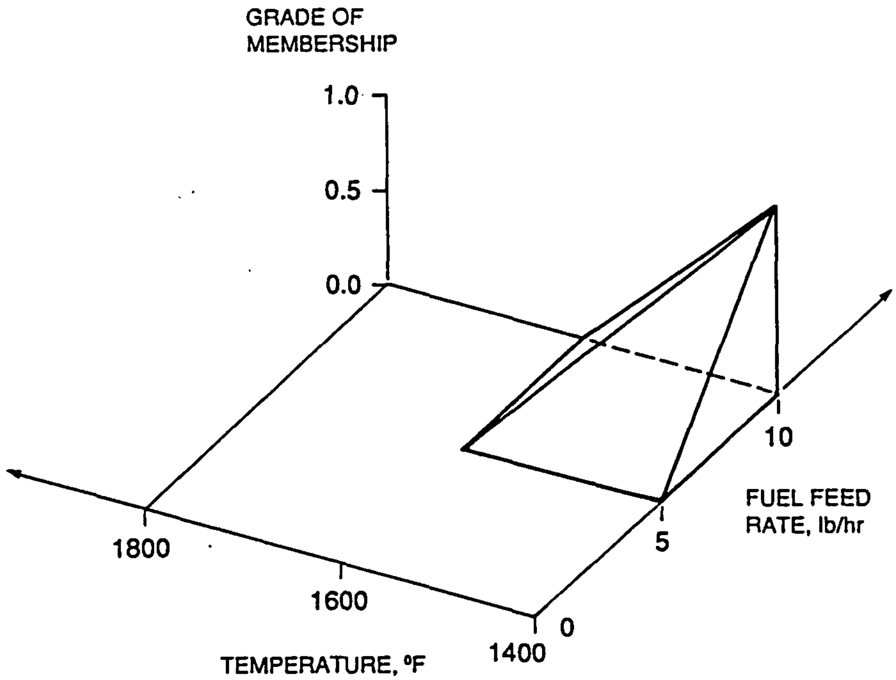

Figure 2.3: Cartesian Product for "Low" Temperature and "High" Fuel Feed Rate of Hypothetical System

In words, the grade of membership for fuzzy relation No. 1 at point (t_i, f_j) in the relation space is the minimum of the grades of membership of "low" temperature for element t_i and "high" fuel feed rate for element f_j. Similar forms of Equation 2.1 are used to separately formulate fuzzy relations for each rule. This equation can be generalized for MIMO systems; more terms are included on the right-hand side of the equation to account for all grades of membership of multiple antecedents and consequents.

After all relations are formed, they are combined to give a complete (overall) relation for the system. This is done by taking the union of the individual relations. To illustrate this point, the three fuzzy relations[2] for the hypothetical system are

[2]Fuzzy relations for the remaining two rules can be formed as was done for the second rule.

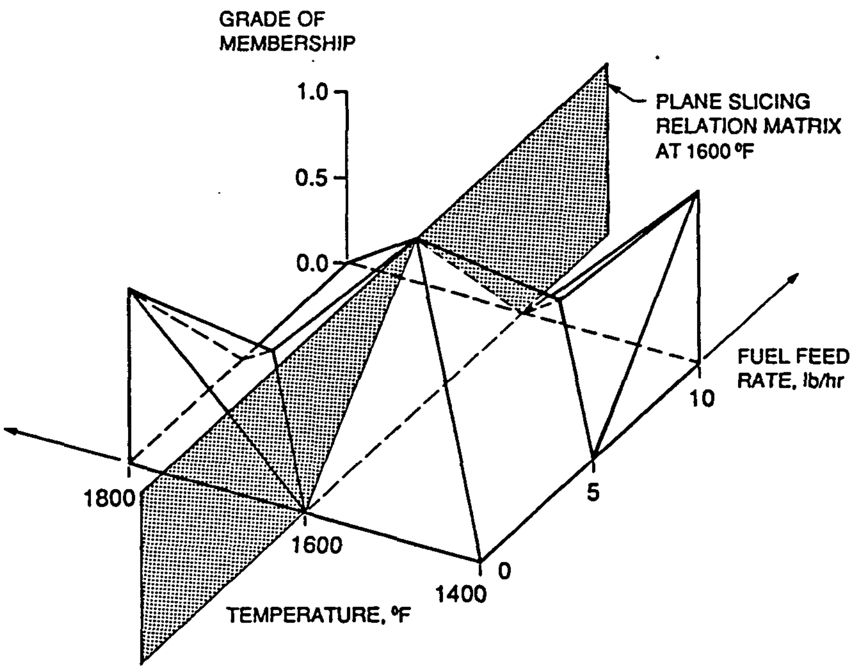

Figure 2.4: Complete Fuzzy Relation for Hypothetical System

"unionized" to give the total relation as shown in Figure 2.4. This surface is the graphical depiction of what is known as the relation matrix. It was generated by the equation:

$$\mu_R(t_i, f_j) = \max[\mu_{R_1}(t_i, f_j), \mu_{R_2}(t_i, f_j), \mu_{R_3}(t_i, f_j)]. \tag{2.2}$$

The overall grade of membership at point (t_i, f_j) is the maximum of all individual fuzzy relations. Equation 2.2 includes as many terms in the right-hand side as there are number of rules. The relation matrix is analogous to the transfer function of a conventional controller.

2.4.3 Inference process

After the relation matrix is generated, it is operated upon in a two-step process to infer control actions. First, Zadeh's compositional rule of inference, an approximate extension of *modus ponens* [6], is used to find fuzzy output sets for fuzzy input sets. Secondly, because each output set has many elements, it is defuzzified to obtain a unique control action. Zadeh's compositional rule of inference is examined first.

In general, Zadeh's rule generates fuzzy output sets for fuzzy input sets. However, if precise inputs are known, then fuzzy singleton inputs [11] are used to infer control actions. Fuzzy singletons are sets having only one element with full membership[3]. This single element is a precisely measured process variable, such as temperature for the hypothetical process. If process variables can not be exactly quantified, fuzzy input sets are then used. Note that the output from the inference process will be fuzzy regardless if the input is fuzzy or not.

To illustrate inferencing using fuzzy singletons, consider the relation matrix in Figure 2.4 for the hypothetical system. If the control action for 1600°F is desired (a fuzzy singleton), the matrix is "sliced" perpendicular to the temperature axis at 1600°F as shown in Figure 2.4. A fuzzy output set is produced as depicted by the resulting cross section. This fuzzy set is repeated in Figure 2.5; note that it is the same as the "normal" fuel feed rate fuzzy set used in rule 2. In fact, all original fuzzy sets are preserved in the relation matrix. Now, if the control action for 1550°F is desired, the matrix is sectioned at 1550°F. The resulting fuzzy output set is shown in Figure 2.6. It has two plateaus showing the influence of both rules 1 and 2.

[3]These sets are degenerate fuzzy sets which can be considered as normal sets.

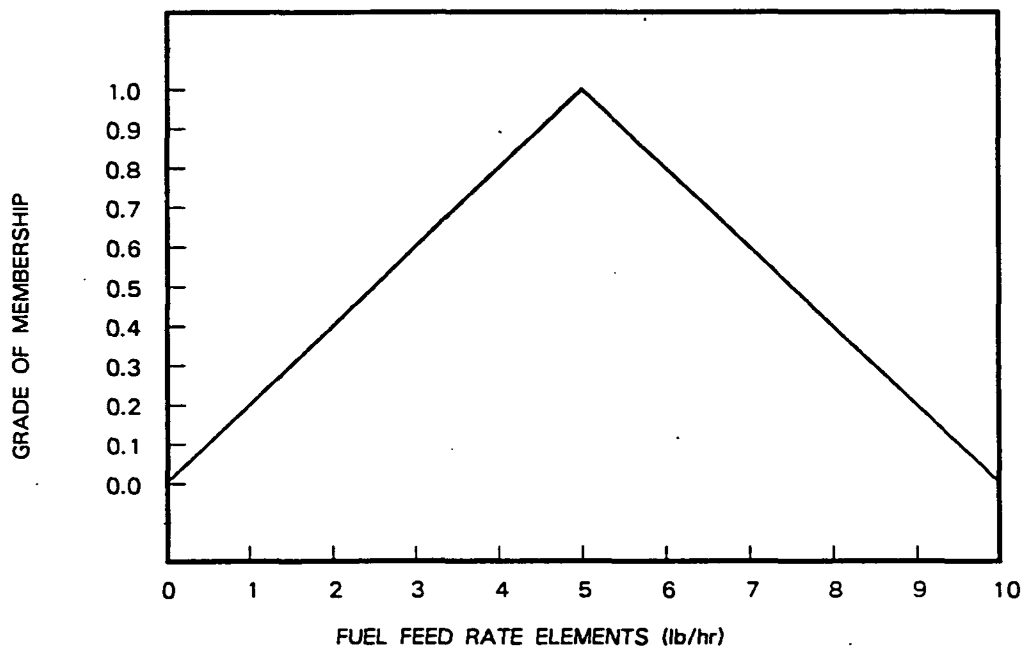

Figure 2.5: Fuzzy Output Set for Hypothetical System for 1600°F Input

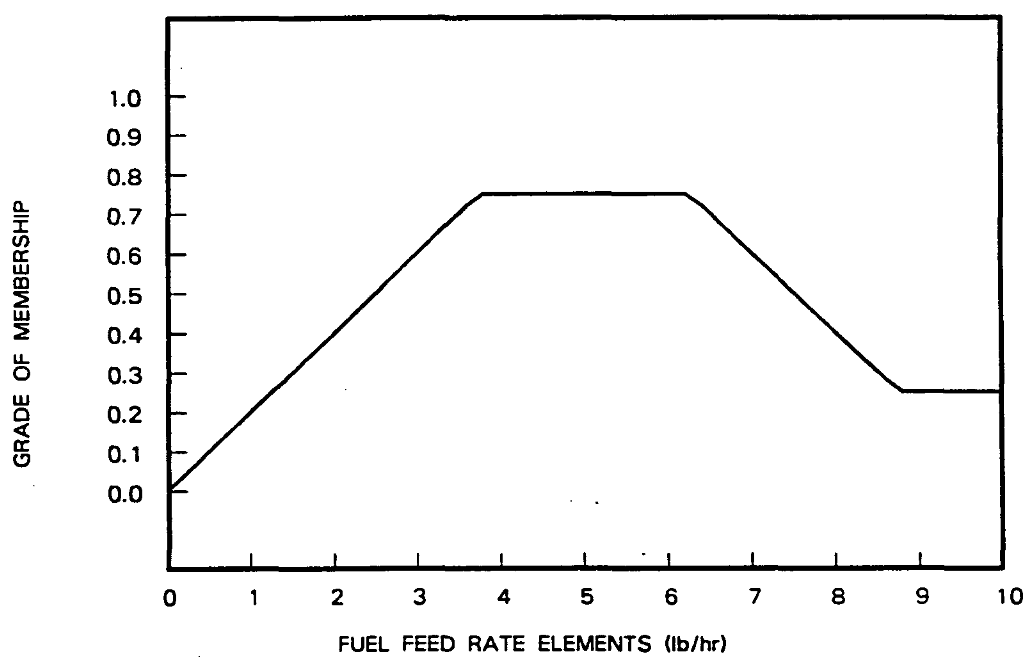

Figure 2.6: Fuzzy Output Set for Hypothetical System for 1550°F Input

The second rule skews the membership function toward the center while the first rule pulls the membership function toward the right. When this output set is defuzzified (as explained later), the control action will lie somewhere between the control actions associated with 100% "low" and 100% "ok". Hence, the inferred action will be seen as an interpolation between the two neighboring rules.

The "slicing" operation is the graphical depiction of Zadeh's compositional rule of inference for fuzzy singleton inputs. In general, Zadeh's rule[4] for the hypothetical system is expressed mathematically as:

$$\mu_F(f_j) = \max_T[\min[\mu_T(t_i), \mu_R(t_i, f_j)]].$$

In words, the grade of membership of the fuel feed rate output set is the largest value over the temperature universe of discourse of the intersection of membership functions for a temperature input set and the overall fuzzy relation. Stated differently, the fuzzy output set is the projection against the fuel feed rate universe of discourse of the 3-D intersection of the temperature input set and the overall fuzzy relation. For all inputs being fuzzy singletons with $\mu_T(t_i) = 1.0$, as would be the case for precise process measurements, the rule reduces to:

$$\mu_F(f_j) = \mu_R(t, f_j) \mid_{t=t_{measured}}$$

where t specifies measured temperature after being "rounded" to the nearest input temperature element of the relation matrix. The reason for rounding is that membership functions, though continuous in theory, are discretized to be manipulated by a digital computer.

[4]The interested reader can refer to [6] for a demonstration of the general form of Zadeh's rule.

To this point, a fuzzy output set has been determined for a fuzzy singleton input. An output set represents many possible control actions. Each action has a possibility (grade of membership for that element or control action) of being applied to the system for that fuzzy singleton. Since the system only accepts non-fuzzy values, the output set is defuzzified — the second step to inferencing — to obtain one control action. Mean-of-maximum (MOM) and center-of-area (COA) are the two main defuzzification techniques most frequently employed.

The mean-of-maximum method selects the control action having the largest grade of membership. In instances where more than one maximum occurs, an average is taken of the multiple maximums. This method only takes into account the strongest elements of the output set. The mean-of-maximum value is computed by

$$\bar{f}_{MOM} = \sum_{k=1}^{N} \frac{f_k}{N}$$

where f_k's are obtained from

$$\mu_F(f_k) = \max_F \mu_F(f),$$

and N is the number of elements having maximum membership.

The center-of-area technique chooses the element having half the area enclosed by the output membership function on one side of the element and half on the other side. All elements of the output set are taken into account by this method. The center-of-area value is computed by

$$\bar{f}_{COA} = \frac{\sum_{j=1}^{n} \mu_F(f_j) \times f_j}{\sum_{j=1}^{n} \mu_F(f_j)}.$$

As indicated by Larkin [14], the center-of-area technique is preferred because it yields better results with smaller control effort than the mean-of-maximum technique.

2.5 Fuzzy Algorithm, Stability, and Tuning

The above sections describe parts of a fuzzy control algorithm that derives actions from heuristic rules. To summarize, all aspects of this algorithm are listed below:

1. Rules and membership functions are obtained from a human expert.
2. A fuzzy relation is generated for each rule by forming the intersection of membership functions of the antecedents and consequents for that rule.
3. An overall relation matrix is formed by taking the union of all individual fuzzy relations.
4. Fuzzy output sets are inferred via Zadeh's compositional rule of inference.
5. Control actions are obtained by defuzzifying fuzzy output sets.

Basically, two methods are used to implement this algorithm. One method calculates actions at each control instant during on-line control. Of the available rules, only those having an effect on the output at a particular control instant are used. Steps 2 and 3 are executed for these rules only, and steps 4 and 5 are then performed in the usual manner. This process is then repeated at subsequent control instants. If controller rules and parameters are static, the other method can be used which calculates actions for selected fuzzy singletons prior to controller usage. These actions are placed in look-up tables for referencing during on-line control. Variants of these basic implementations can also be employed. Larkin [14] presents a computationally efficient algorithm that combines steps 2-4. After an implementation is chosen for the system under study, questions of stability and tuning must be addressed.

Conventional control engineering imposes a certain rigid design approach which must include stability analysis. Because fuzzy equivalents of typical criteria used in conventional design procedures are still in the developmental stages, no good way exists to predict whether a system will be stable. However, if the system was stable during manual control, the system should be stable under fuzzy control assuming all significant human knowledge about the system is accurately represented in the rules.

As for tuning, King and Mamdani [8] state that it can be accomplished in three different ways:

1. modify control rules,
2. change discretization levels of the membership functions, and
3. change definition of the linguistic variables — modify the membership functions.

The second method needs explaining. In theory, membership functions are continuous; but, in practice, they must be discretized to be manipulated by a digital computer. Size of the discretization levels can have a profound affect on system stability. If discretization levels are too large, calculated control actions look like those from a multi-level relay with coarse steps. Hence, the system may limit cycle.

King and Mamdani found the second method to be the most effective for tuning. However, Tong [15] argues via a relation matrix analysis that the best way to tune a fuzzy controller is to modify the control rules. He states that items 2 and 3 should be chosen to match qualitative assumptions made about the process and should not be changed unless the assumptions prove incorrect.

Computer simulations prove useful in determining system stability and performance. However, those systems that are too ill-defined to apply classical control techniques may also be extremely difficult to model with reasonable certainty. In these instances, stability analysis and tuning are best left to analysis of actual test data. A trial-and-error approach may also be used in making tuning decisions.

3. EXPERIMENTAL APPARATUS AND PROCEDURES

This section describes the fluidized bed combustor and its manual operating procedures, data acquisition equipment, and control hardware.

3.1 Fluidized Bed Combustor

Essentially, a fluidized bed combustor consists of a chamber filled with particles, such as sand, through which a stream of air is passed vertically upward [16]. If the air stream is at sufficiently high velocity, the drag force acting on the particles balances the bouyancy-adjusted weight of the particles; in this state the bed is said to be fluidized. Fluidized beds consist of an emulsion phase and a bubble phase. The emulsion phase is a homogeneous mixture of sand and air. The bubble phase consists of discrete volumes of air which are heterogeneously dispersed throughout the emulsion phase. Bubbles represent air flow in excess of what can be transported by the emulsion phase.

Fluidized bed combustors are characterized by coupled dynamics and severe nonlinear operation due to fluidization, radiative heat transfer, and chemical kinetics. Figure 3.1 illustrates the nonlinear dependence of convective heat transfer coefficients on fluidization velocity [17]. Air velocities through the combustion bed must be greater than the minimum fluidization velocity, U_{mf}, required to fluidize the bed.

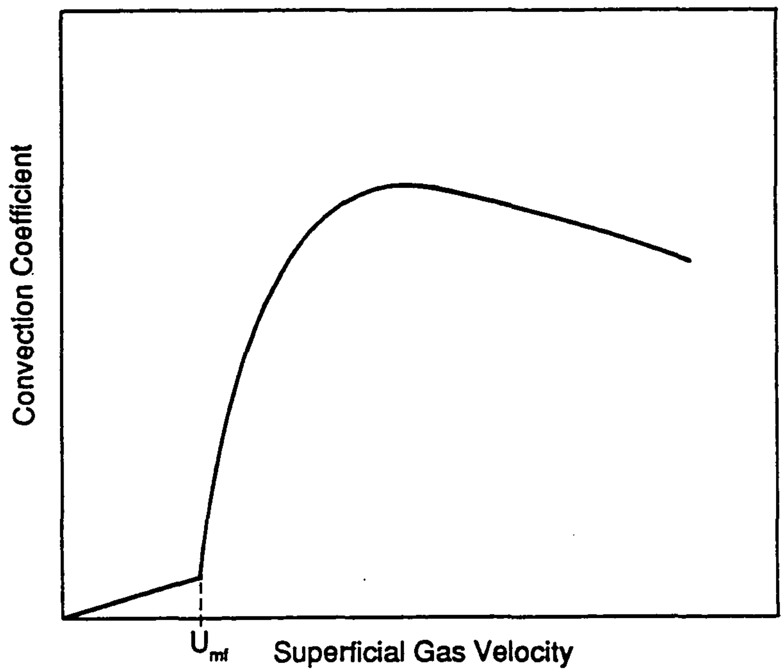

Figure 3.1: Dependence of Convective Heat Transfer Coefficients on Air Velocity

Also, air velocities should not be so great that small fuel particles elutriate from the combustor.

The fluidized bed combustor used in this research is of a novel two-bed design [17]. Figure 3.2 schematically illustrates the water-jacketed, cylindrical fluidized bed. The central combustion bed is provided with air through a circular distributor plate from an air plenum. Coal is fed by auger into the combustion bed at a rate determined by the desired heat generation rate. If fluidization requirements are satisfied, air flow into this bed (known as primary air) is typically set at a level 20% greater than stoichiometric requirements to ensure good combustion. The annular heat transfer bed is supplied with air (known as secondary air) from another plenum. The annular bed is surrounded by a water jacket that removes heat from the combustor in the form

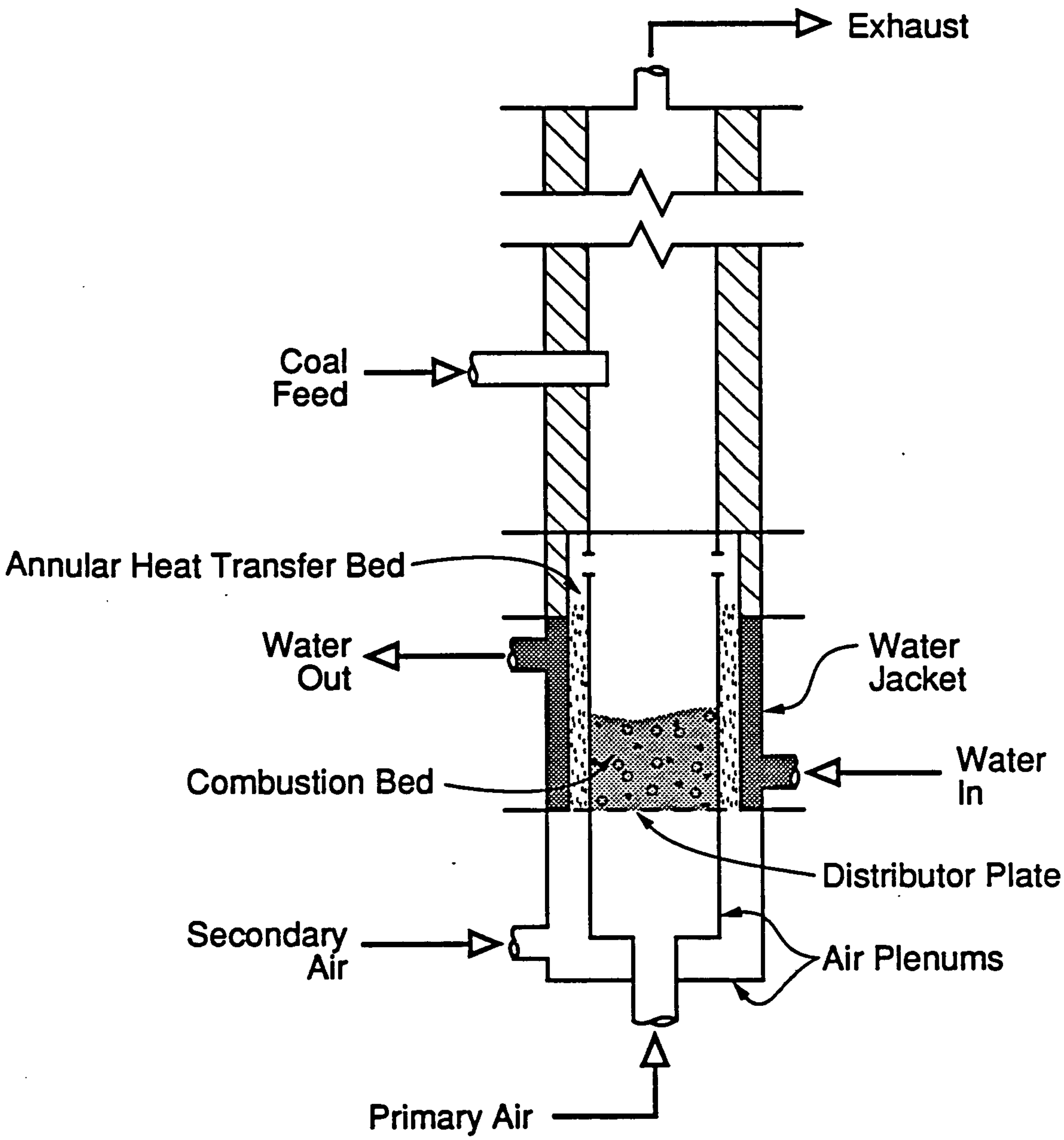

Figure 3.2: Combustor Schematic

of hot water. The annular bed acts as a thermal transistor. A variable secondary air flow rate controls the amount of heat released across the annular bed width to the water jacket thereby providing a means for controlling combustion bed temperature.

Control inputs for this system are fuel feed rate, primary air flow rate, and secondary air flow rate. System outputs are combustion bed temperature and composition of gases in exhaust indicating combustor efficiency. Of particular importance in this study is the concentration of oxygen in the flue gases. A reading of 3.5% oxygen is desired because it implies complete combustion of the test coal with 20% excess air.

3.2 Manual Operating Procedures

A particularly challenging task is to bring the combustor up to a nominal operating point from ambient conditions. The combustor is first preheated to 1200°F by burning liquid-petroleum (LP) gas in the combustion bed. When this temperature is achieved, coal feeding commences. LP gas flow is terminated at 1450°F — a point where coal combustion is self-sustaining. During preheating, a nominal secondary air flow rate, allowing for a minimally fluidized annular bed, is initiated to reduce thermal stresses in the cylindrical wall separating the combustion and annular beds. A packed annular bed would restrain transverse thermal expansions of the inner combustor wall causing high stress levels to develop.

When coal feeding begins at 1200°F, coal feed rate is not immediately set to its desired value. Instead, it is increased in a step-wise fashion over a period of time. The reason for this incremental approach toward a steady-state feed rate is two-fold. First, FBCs exhibit time lags between entrance of fuel into the combustor and combustion of that fuel. Second, combustor efficiency is degraded by large step changes in fuel feed rate due to incomplete combustion.

An effective approach used in manual FBC start-up is to give small increases in feed rate followed by measurements of oxygen in the flue gas. A decrease in oxygen indicates that fuel is being combusted. After an appropriate oxygen decrease, primary air flow rate is increased, and fuel feed rate is again stepped up. Air flow rate is increased only after flue gas oxygen has decreased because an increase in fluidizing air will cause significant combustion bed temperature drop (due to advective cooling) before heat can be generated by the combustion process. Primary air flow is ultimately adjusted to achieve 3.5% flue gas oxygen since this value implies combustion with 20% excess air.

Unfortunately, 3.5% flue gas oxygen is not always attainable because the manual operator must maintain primary air flow rate within certain limits. For low fuel feed rates, primary air flow rate required to maintain 3.5% flue gas oxygen may not be high enough to keep the combustion bed fluidized. This bed, unlike the annular bed, must always be fluidized to ensure proper combustor operation. Hence, primary air flow is not allowed to fall below 9 scfm, which can, for certain feed rates, make oxygen percentage high. For high fuel feed rates, the required primary air flow rate for 3.5% oxygen is so high that combustion bed particles are blown from the combustor. Therefore, primary air flow is normally maintained below 40 scfm, which can make oxygen percentage low.

Temperature of the combustion bed is maintained at 1600°F by adjusting secondary air flow. This temperature is chosen for steady-state operation because sulfur dioxide absorption by limestone sorbent is maximized near this temperature [18]. For low fuel feed rates, the rate of heat generation may not be sufficient for the bed to reach 1600 °F. In these cases, secondary air is turned off after preheating so that the combustor may reach its highest suboptimal temperature.

3.3 Test Equipment

A schematic of the computer-controlled FBC is depicted in Figure 3.3. Data acquisition and digital control are preformed by a Zenith Z-158 microcomputer. The Z-158 is based on an Intel 8088 CPU operating at 4.77 Mhz. The unit is configured with an Intel 8087 math coprocessor, 640 Kb of addressable memory, and a 20 Mb hard drive. Both data acquisition and digital control codes are written and compiled in Microsoft QuickBASIC 4.0 (see Appendix A).

3.3.1 Data acquisition hardware

All analog signals from measuring devices are interfaced to the microcomputer using a Metrabyte DAS-8 board. The DAS-8 is an 8-channel A/D converter and timer/counter interface. The A/D converter is a 12-bit successive approximation converter with conversion times of 25 microseconds.

Temperature data are obtained from three K-type (chromel/alumel) thermocouple probes in the central bed, and one K-type thermocouple probe in the annular bed. To protect them from the abrasive bed environment, thermocouples are enshrouded in a 304 stainless steel casing. The probes are connected to a Metrabyte

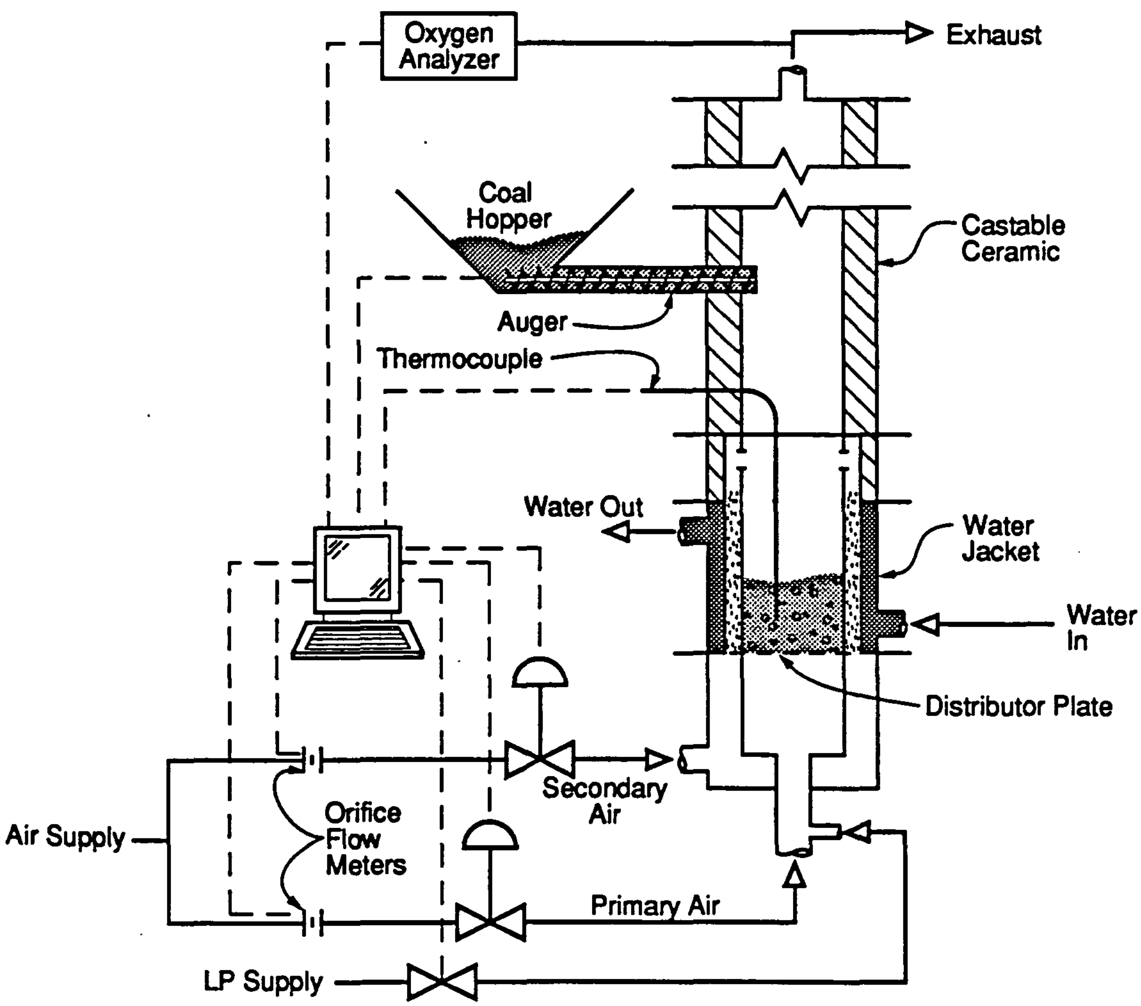

Figure 3.3: Schematic of Computer-controlled FBC

sub-multiplexer board (Model EXP-16), which amplifies thermoelectric voltages and provides cold junction compensation.

Both primary and secondary air flow rates are calculated from pressure drops across orifice flow meters. Pressure drops are measured with a Schaevitz LVDT pressure transducer (Model P3061). Air flow rates are calculated by linearly interpolating from calibrated pressure drop data.

Exhaust gases are monitored by several Horiba and Beckman Industrial analyzers for oxygen, carbon monoxide, carbon dioxide, nitrogen oxides, and sulfur dioxide concentrations. Oxygen is the only gas of interest in this investigation; it is measured using a Beckman Industrial paramagnetic oxygen analyzer (Model 755).

3.3.2 Control hardware

All variable-control hardware is commanded by 0-5 volt signals provided by the computer through a Metrabyte DDA-06 board. The DDA-06 is a 12-bit, 6-channel analog output interface with 24 parallel digital I/O lines. Each D/A channel can source 5 mA of current, but a maximum of 8.13 mA is required by the E/P transducers. To boost current to required values, a current-amplifying circuit is placed between the control hardware and the computer. The circuit was constructed using op-amps configured as unity-gain voltage amplifiers [19] with input offset voltage null circuitry [20]. Each amplifier produces a maximum of 25 mA output.

Primary and secondary air flow rates are adjusted by Fisher Design GS valves with Fisher Type 513R reversible pnuematic actuators. Valves, configured for fail-close operation, are regulated by Bellofram Type 1000 E/P Transducers. For each valve assembly, a transducer produces 3-15 psig of pressure causing the actuator diaphragm and valve stem to move against a return spring within the actuator body. Variable flow is produced by changing the transducer's pressure output. Calibrated data are used to produce desired volumetric flow rates. However, considerable flow hysteresis attributed to friction in the valve stem packing prevented reliable calibration of the original valve assemblies.

To remove flow hysteresis, position feedback control loops were installed on the valves. Valve stem positions are sensed by voltage drops across TCI linear potentiometers (50 kΩ, 2" stroke) strapped to the valves. Conventional PI controllers, patched on a EAI TR-20 analog computer, position the valve stems according to calibrated reference signals to obtain repeatable, desired flow rates. The controllers were tuned by the Ziegler-Nichols transient-response method [4].

Coal (Illinois No. 5 Rapattee) is fed into the combustor using an AccuRate dry chemical feeder (Model 602). The auger feeder is equipped with a control board that accepts an input signal from the DDA-06 to adjust auger speed. The feeder was calibrated for feeding 3/8 in. × 8 mesh coal in terms of pounds per hour.

LP gas flow into the combustor is on-off controlled by an Atkomatic solenoid valve (Dymo 15400-x). To obtain a certain gas flow rate into the combustor, a Cole-Parmer rotameter (tube #FM044-40ST, 316 SS float) is inserted after the solenoid valve to restrict the flow. LP gas is ignited using two spark electrodes powered by a Webster ignition transformer (Type 312-25AX0202). Both the solenoid valve and electrodes are energized by a computer-controlled electric outlet panel. Each electrical port is controlled by a relay from a Metrabyte ERA-01 8-channel relay board external to the computer. The ERA-01 is commanded by a Metrabyte PIO-12 board, a 24-bit parallel digital I/O interface board.

4. FUZZY LOGIC CONTROLLER DESIGN

This chapter describes the fuzzy controller in its final form. The final version represents an evolution from "drawing board" to actual system. The original controller was first tested by computer simulation to verify stability and performance; details are given in [21] and Appendix B. Simulations showed good controller performance after changes were made to effective secondary air flow gains. When the fuzzy controller was installed on the actual system, it became apparent that these changes were incorrect for the actual combustor. The model erroneously predicted temperature changes due to secondary air flow adjustments. Tuning of the controller, using the methods listed in Section 2.5, was necessarily completed on the actual system. The information in this chapter reflects changes made to the original controller; changes are mentioned when appropriate.

4.1 Control Objectives

The combustor in this investigation is primarily used to study fluidized bed combustion processes. Combustion tests are conducted at steady-state conditions for different fuel feed rates. To facilitate these tests, two main control objectives are desired. One objective is to bring the combustor up to 1600°F from ambient conditions and to maintain this temperature to ensure optimum sulfur dioxide absorption by the

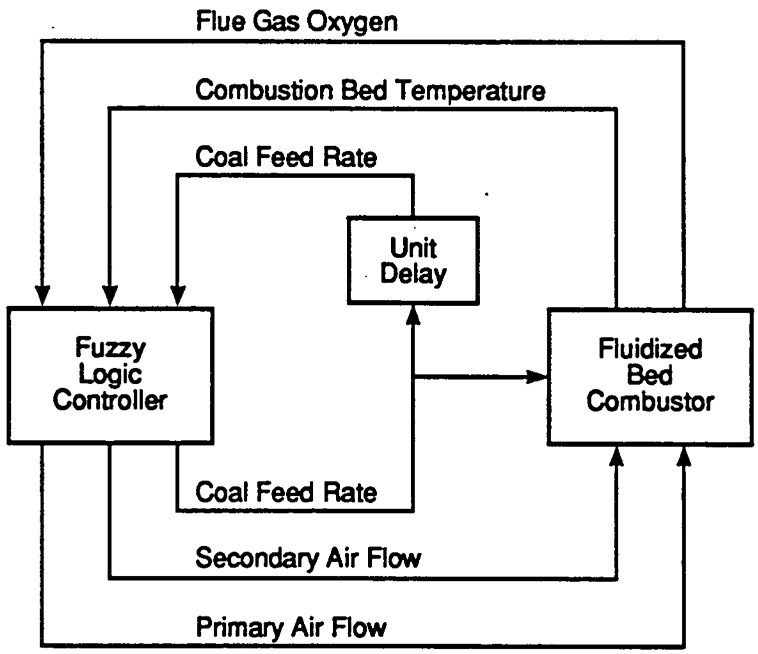

Figure 4.1: Block Diagram of Controlled System

limestone sorbent. The other objective is to maintain 3.5% oxygen in the flue gas to ensure good combustor efficiency. These objectives need to be achieved independent of fuel feed rate.

4.2 Control Rules

Figure 4.1 is a block diagram of the controlled system. Note that fuel feed rate is present in both the input and output of the controller. This is done to implement the incremental firing rate approach. After the desired fuel feed rate is obtained, no more control actions occur for this loop. From this point on, only adjustments are made to combustion bed and annular bed air flow rates to affect changes in flue gas oxygen concentration and combustion bed temperature, respectively.

Control rules were acquired by observing FBC operation and interviewing the human operator[1]. Three different sets of control rules were obtained for this combustor — one set each for controlling temperature, flue gas oxygen, and coal feed rate. Because temperature and flue gas oxygen have constant setpoints, 1600°F and 3.5%, respectively, an error signal for each of these variables is not necessary to drive the controller. Instead, control actions prescribed for these variables are based upon their respective states. However, fuel feed rate is expected to have many different setpoints. Therefore, an error signal is generated for this variable. The form of each rule set is depicted below:

- If temperature = _____ and temperature change = _____, then secondary air flow rate adjustment = _____.
- If oxygen = _____ and oxygen change = _____, then primary air flow rate adjustment = _____.
- If oxygen = _____ and coal feed rate error = _____, then coal feed rate adjustment = _____.

Output of the control rules represents changes to be made to control settings of the combustor. This type of output is referred to as incremental control. At any control instant, the value from the rules is added to previous values to obtain the value that gets passed to the combustor. An incremental approach is used because the human operator of the FBC was only able to state approximately how much he would change combustor inputs for given combustor states. An added benefit of using

[1]The operator was a graduate student with 4 years experience running a lab-scale FBC.

incremental rather than absolute control is that control actions are summed to achieve zero steady-state error. Thus, computational complexity of the controller is reduced by eliminating the need of *sum-of-error* as an input to obtain zero steady-state error for an absolute control scheme.

As an interesting side note, a major difference between the final controller and original controller in [21] is that the original controller did not use changes in temperature and oxygen. The original controller was intentionally kept simple to easily study certain aspects of the system. For this simple controller, it was found that discretization levels were initially chosen so large that limit cycles developed. While this controller was helpful for selecting discretization levels (more about this later), it proved too fragile to be of practical use. By not utilizing temperature and oxygen changes, information is lost about the system that is necessary for good control. For example, the simple controller increases secondary air when temperature is above the setpoint. This sounds reasonable, but temperature may already be dropping. Increasing secondary air at this point may cause temperature to drop below the setpoint. This controller behavior could make the combustor unstable. Therefore, changes in temperature and oxygen are used as inputs in the final controller.

Before the operator could be quizzed about his actions, the questions had to be formulated by asking the operator to describe inputs to the above rule structures. Using these descriptions as fuzzy set names of the antecedents, questions for a particular rule set were generated by using all possible combinations of these descriptions. The operator characterized combustion bed temperature and flue gas oxygen as either "low", "ok", or "high". Temperature change, oxygen change and coal feed rate error were identified as "positive", "near zero", or "negative". While "near zero" evokes images of fuzzy sets, "positive" and "negative" do not; they sound like descriptions of regular sets. However, their meanings and definitions are fuzzy. "Positive" and "negative" could be thought as "positive large" and "negative large", respectively.

The operator was then asked to list his actions for each question. The actions were given as "more", "none", or "less" for increments in control settings. These labels are used to describe changes in all controller outputs, but "more", for example, has a different meaning for changes in primary air than it does for secondary air.

As an example of the rules, the rule set for temperature control is examined here. Temperature is described as "low", "ok", or "high"; temperature change is described as "positive", "near zero", or "negative". Given three states for temperature and three states for temperature change, the total number of combinations of these states is nine; hence, nine rules exist.

Rules with operator's responses are given in "truth table" format in Table 4.1. Some rules have the same response for similar antecedents. Take, for example, entries 2 and 3 in Table 4.1. They can be combined into one rule by defining the new fuzzy set **NOT POSITIVE**[2] for temperature change. This new set is simply the complement of **POSITIVE**. The new rule reads as:

> If *temperature* = **LOW** and Δ *temperature* = **NOT POSITIVE**, then
> Δ *annular bed air flow* = **LESS**.

Thus, this rule set can be reduced. The process is analogous to Karnaugh mapping [22] used in digital circuit design. Size reduction of rule sets is not always possible; it

[2]In this section, fuzzy sets are denoted by bold-faced names and state descriptions are denoted by quoted names.

depends upon the operator's responses. Whenever one antecedent changes between two rules and no corresponding change occurs in the consequents between the rules, reduction is possible. The reduced temperature set and all other rule sets are listed in Appendix C, using the same format as in Table 4.1.

Table 4.1: Truth Table of Temperature Control Rules (Full Set)

T_{BED}	ΔT_{BED}	$\Delta Sec.\ Air\ Flow$
LOW	POSITIVE	NONE
LOW	NEAR ZERO	LESS
LOW	NEGATIVE	LESS
OK	POSITIVE	MORE
OK	NEAR ZERO	NONE
OK	NEGATIVE	LESS
HIGH	POSITIVE	MORE
HIGH	NEAR ZERO	MORE
HIGH	NEGATIVE	NONE

As a final note, an alternate presentation of the rules is possible that allows a more computationally efficient fuzzy algorithm to be employed. For each fuzzy description of incremental controller output, one rule can be formed. For example, all of the rules giving **NONE** as incremental output can be combined into the following rule:

If (*temperature* = **LOW** AND Δ *temperature* = **POSITIVE**) OR

(*temperature* = **OK** AND Δ *temperature* = **NEAR ZERO**) OR

(*temperature* = **HIGH** AND Δ *temperature* = **NEGATIVE**),

then Δ *annular bed air flow* = **NONE**.

The connective ORs are treated in the same manner as the ANDs, i.e., ORs represent minimum operators. Rules represented in this format can be manipulated more efficiently since the minimum operation of the rule's consequent is done only once to the collective intersection of the rule's antecedents rather than being done individually for the three previous rules.

The operator now needs to explain what he considers "low", "ok", "high", etc. This next step is the formation of membership functions for all fuzzy sets.

4.3 Membership Functions

For simplicity, only membership functions of fuzzy sets describing temperature are derived here. However, this derivation technique was applied to all other membership functions. To generate temperature membership functions, the operator was asked what was 100% "low", 100% "ok", and 100% "high" as well as what was 0% "low", 0% "ok", and 0% "high" for temperature. Because humans tend to think linearly when comparing magnitudes, fuzzy sets were generated by drawing straight lines through 100% to 0% for each state.

In actuality, **OK** temperature was determined first. **LOW** and **HIGH** were then derived from **OK** such that **LOW** became the complement of **OK** on the low side of 100% "ok", and **HIGH** became the complement of **OK** set on the high side of 100% "ok". Figure 4.2 illustrates this point. The desired temperature is 1600°F, which is 100% "ok" temperature. Temperature could drop to 1525°F before being absolutely unacceptable. Similarly, temperature could rise to 1675°F before being completely intolerable. The 1525°F and 1675°F are the 0% end points for **OK**. Once the end points for **OK** were selected, all temperature membership functions easily followed. All membership functions for fuzzy sets used in the rules appear in Appendix D.

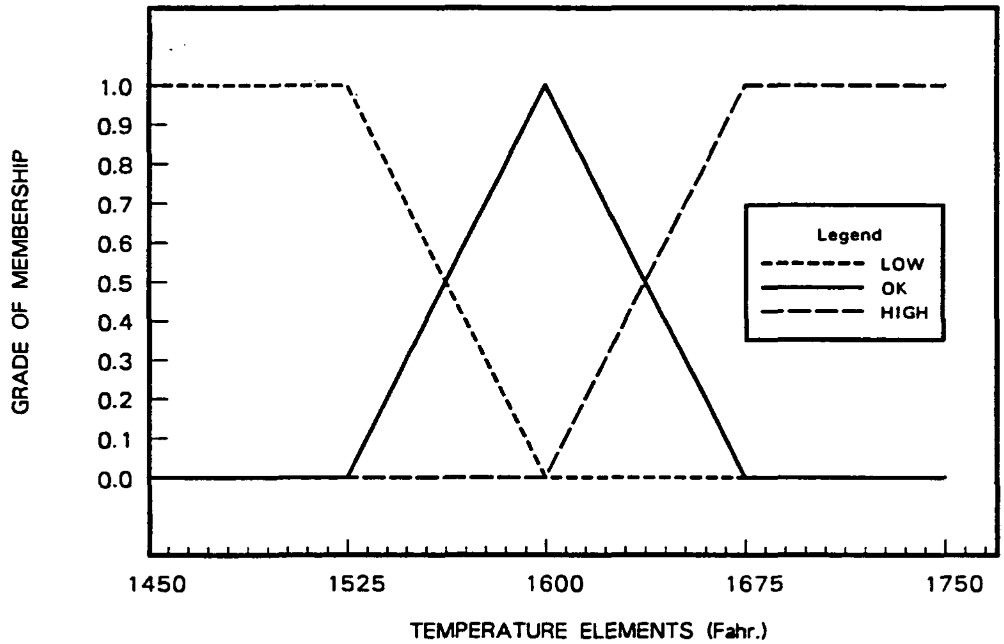

Figure 4.2: Relation between Membership Functions for "Low", "Ok", and "High" Combustor Temperature

The next step in fuzzy controller design is to determine discretization levels of membership functions. In this investigation, the FBC is controlled by referencing control actions in look-up tables. After control actions are calculated using fuzzy logic inferencing, they are placed in look-up tables. Therefore, discretization level considerations are based on a look-up table implementation. Otherwise, if actions were to be calculated at each control instant, other considerations would be warranted.

By preprocessing control actions, discretization of output sets of the rules can be made as fine as needed to achieve good control without increasing data in the look-up tables. This is true because only one defuzzified action exists for each set of inputs to each table. However, computation time to form the tables does increase. Recall for controlling temperature, adjustments are made to annular bed air. Fuzzy sets describing these adjustments have discretization intervals of 0.5 scfm. This value represents the coarsest level that still allows fine tuning of temperature.

Small discretization levels in output sets will not be effective if discretization levels of input sets are too large. By decreasing discretization levels of fuzzy input sets, data (number of entries) in the look-up tables increase. Although finer control is produced for smaller discretization levels, formulation time and size of the look-up tables can become unacceptably large. Thus, discretization levels of fuzzy input sets must be judiciously chosen to keep the amount of data manageable while maintaining acceptable control. The criterion used for selecting the discretization levels for temperature fuzzy sets is based upon how well control was desired. For these fuzzy sets, a maximum discretization level of 10°F[3] was chosen. Although temperatures greater than 10°F away from the setpoint can be tolerated, larger discretization levels were found to produce limit cycles. All discretization levels, as denoted by tick marks, accompany their respective membership functions in Appendix D.

4.4 Look-up Tables of Control Actions

With all fuzzy controller design steps completed, a look-up table was formed for each controller output. The procedure included forming a relation matrix for each individual rule. An overall relation matrix was then formed by taking the union of all individual relation matrices. Zadeh's compositional rule of inference was used to find fuzzy output sets for fuzzy singleton inputs which are simply discrete elements of the input fuzzy sets. Fuzzy output sets were defuzzified by using the center-

[3]Because of convenience, intervals of 7.5°F were actually used.

of-area technique. This method was chosen over the mean-of-maximum technique because of its reported superiority [14]. The above steps were programmed on a VAX 11/785 mainframe in FORTRAN[4] to facilitate simulations. However, defuzzified control actions were placed in look-up tables for use on a Zenith Z-158 microcomputer used for control.

Look-up tables were formed by assigning arrays in QuickBASIC 4.0 control actions for given fuzzy singleton inputs where the inputs form the array indices. Control actions were referenced from the arrays by using process measurements. Because array indices can only have integral values, the measurements were "rounded" to the nearest array index before reading a control action. Also, if an input "falls" outside of the space for which actions are defined, the nearest array index is used. This has the effect of producing saturation in controller increments but not in total controller action since it can increase at the next control instant.

4.5 Special Implementation Considerations

Not all of the human operator's actions are imprecise. Because of this reason, conventional logic, rather than fuzzy logic, is employed to control precisely defined operations in this application. LP gas flow is on-off controlled — being turned on at ambient conditions and turned off at 1450°F. A minimum air flow rate is maintained in the combustion bed to ensure that it is always fluidized; proper combustor operation requires a fluidized bed. Secondary air flow is initially set to a nominal rate to avoid thermal stresses in the two-bed geometry.

[4]Sample FORTRAN code for generating temperature control actions appears in Appendix A.

A non-fuzzy adjustment to primary air is made when LP gas is turned off. Flue gas oxygen rises rapidly when LP gas flow is terminated due to less fuel being combusted. Shutting off LP gas is a great disturbance to the system that is unparalleled by any other disturbance. The fuzzy controller was not initially designed to handle this "outlier" disturbance. Although the fuzzy controller could have been redesigned to contend with this one-time disturbance, a quicker and simpler method is to use conventional logic to reduce primary air flow when LP gas is shut off. This is done by remembering the primary air flow rate before coal is added; this amount of air flow is needed to burn LP gas with approximately 3.5% oxygen in the flue gases. When LP gas is shut off, the primary air flow rate is reduced by this remembered air flow to approximately cancel the disturbance's effects.

Another special consideration for controller design is that of secondary air flow. As illustrated in Figure 3.2, secondary air flow is channeled into the combustor freeboard, where samples of flue gas are taken for the percentage analysis, just above the combustion bed. Therefore, changing secondary air will affect flue gas oxygen percentage as well as combustion bed temperature. Because combustion takes place in the freeboard as well as the combustion bed, having oxygen added above the combustion bed is not objectionable. However, the addition of secondary air to the freeboard is a coupling of dynamics which can degrade controller performance. To decouple flue gas oxygen from changes in secondary air flow, increments in secondary air commanded by the fuzzy controller are subtracted from those of primary air. A coupling of temperature to primary air flow also exits[5], but its dynamics are not

[5]Changing primary air flow will change convective heat transfer coefficients in combustion bed which will affect combustion bed temperature.

simply additive, like they are for secondary air, thereby making decoupling difficult. Also, the coupling effects are not as pronounced as those for secondary air. For these reasons, temperature was not decoupled from primary air flow.

The last consideration in controller design is sampling time. As in any sampled-data system, sampling time affects eigenvalues of the system. Also, sampling time, as well as membership functions, affects effective gains of the fuzzy controller using incremental output. The human operator said he would enact new control actions in intervals anywhere from several seconds to several minutes. This specification is too vague — fuzzy controllers, like conventional controllers, need definite sampling times. To eliminate this ambiguity, the combustor was operated under fuzzy control with the operator deciding when new control actions were to be applied. Determining the sampling time is best described as a cognitive process. When the operator perceived sufficient system response to previous actions, new actions were given. This type of behavior is reminiscent of a "wait-and-see" controller [23]. A sampling time of 20 seconds was selected; but to avoid aliasing of the change-in-temperature signal, temperature change was calculated over a 5-second interval prior to the control instant. Hence, the control system uses multirate sampling.

5. RESULTS AND DISCUSSIONS

Results presented in this section are from two different control methodologies. First, fuzzy logic control is examined; classical PI control is then used as a benchmark for evaluating the fuzzy controller's performance.

Combustor start-up operation under fuzzy control is depicted in Figures 5.1, 5.2, and 5.3. The goal is to bring the combustor temperature up to 1600°F while maintaining 3.5% oxygen in the flue gases for a coal feed rate of 15 lb/hr. This feed rate represents a worst-case scenario since the combustor is operating near full load.

Figure 5.1 shows combustion bed temperature and secondary air flow. Although not shown in the figure, temperature rises from an ambient 70°F to 1200°F from solely burning LP gas in the combustor. Recall that combustor temperature is initially brought up to 1200°F before coal is added; at 1200°F coal feeding commences (see Figure 5.3). When temperature reaches 1450°F, LP gas flow is terminated because coal combustion is autogeneous at this temperature. As can be seen, a slight drop in temperature is realized causing the fuzzy controller to reduce secondary air flow from its initial nominal setting. Temperature quickly resumes its upward ascent as secondary air flow is reduced and coal consumption increases.

At approximately 21 minutes, temperature overshoots its setpoint of 1600°F. Eventually, temperature is driven downward to the setpoint by increases in secondary

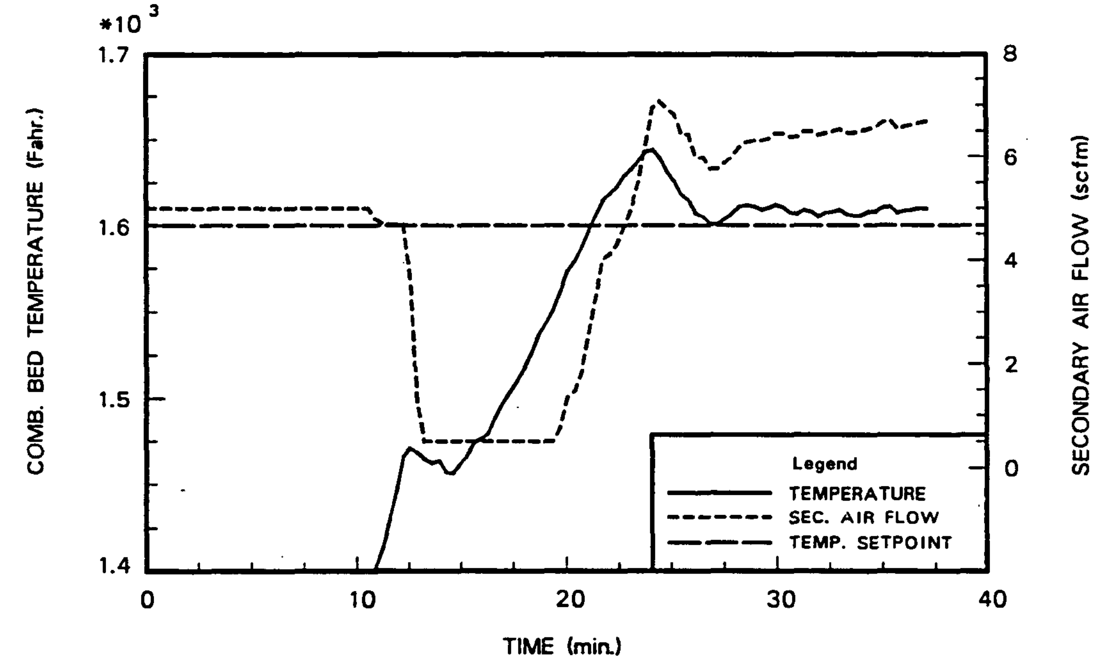

Figure 5.1: Combustion Bed Temperature and Secondary Air Flow Under Fuzzy Control for 15 lb/hr of Coal

air flow. Although temperature reaches 1600°F, its steady-state value settles about 10°F above this setpoint; this is primarily attributed to controller inaction. When temperature and temperature change become small enough, the controller makes no adjustments. This is caused by the discrete nature of the look-up tables of control actions. Recall that each variable is rounded to the nearest entry in the look-up tables before accessing control actions. Thus, when temperature and temperature change fall within less than half the interval of their respective discretization levels from their desired values, the values are rounded to the desired values thereby causing the controller to "think" it has reached its objective. Finer discretization levels could have been used to force temperature closer to its setpoint, but this was not done because the resulting steady-state error is insignificant and this behavior was only found in this test run.

Figure 5.2 shows variations in flue gas oxygen and primary air flow, and Figure 5.3 depicts coal feed rate. Oxygen percentage starts at an ambient 21% and rapidly decreases when LP gas is ignited. The controller increases primary air to maintain flue gas oxygen at its setpoint of 3.5%. At approximately 8 minutes, oxygen sharply drops below the setpoint due to introduction of coal into the combustor. Feed rate is increased in a step-wise fashion, so as not to severely decrement combustor efficiency, to 15 lb/hr.

The controller responds to decreasing oxygen by increasing primary air flow. After oxygen has sufficiently risen, more coal is added thereby causing another decrease in oxygen. This cycle is repeated until the desired feed rate is reached. When flue gas oxygen is low, increments in feed rate will also be low. As can be seen from Figures 5.2 and 5.3, regions of little or no feed rate increments correspond to low oxygen.

From the time coal feeding is initiated until the time desired coal feed rate is attained, the average of flue gas oxygen has approximately constant error. This has the appearance of a classical Type I system experiencing a ramp input [24]. This is due to the integrating action of the incremental fuzzy controller responding to increases in coal feed rate that are ramp-like.

At approximately 13 minutes, an upward spike occurs in flue gas oxygen corresponding to LP gas shut-off. Primary air flow rate is quickly reduced to bring oxygen back down to the vicinity of its setpoint. After the target feed rate of 15 lb/hr is reached, flue gas oxygen settles near 3.5%. The greatest steady-state oxygen

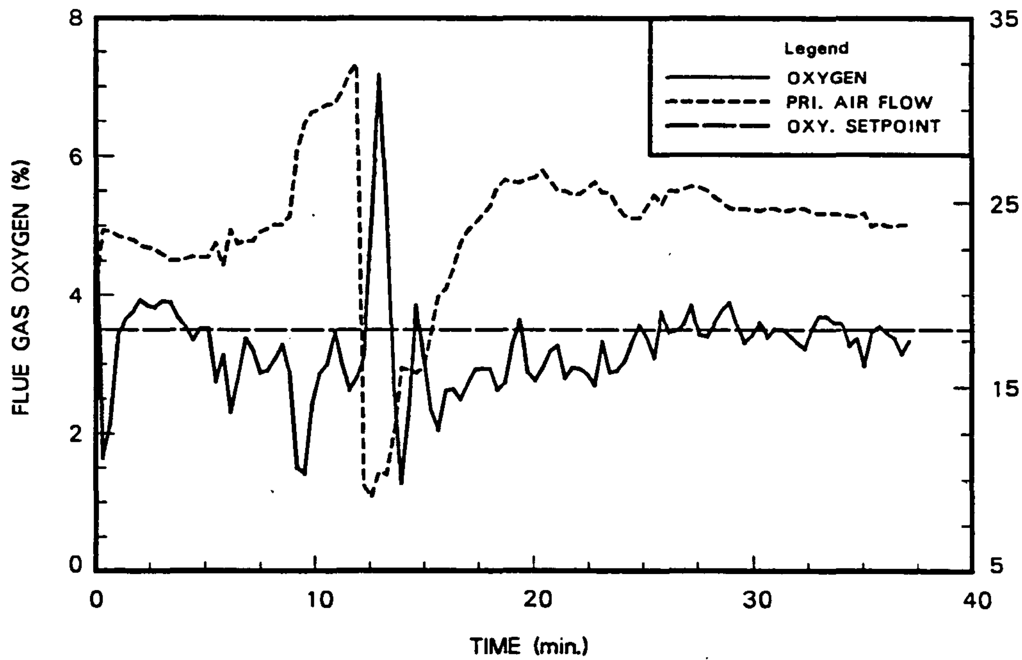

Figure 5.2: Flue Gas Oxygen Content and Primary Air Flow Under Fuzzy Control for 15 lb/hr of Coal

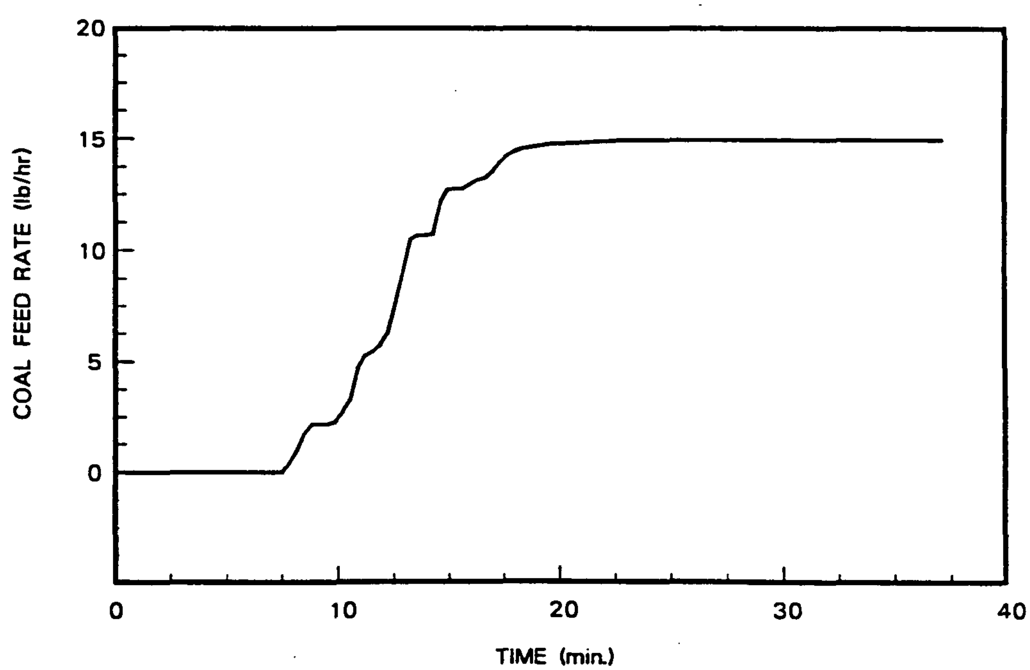

Figure 5.3: Coal Feed Rate Under Fuzzy Control

fluctuation, being within tolerable ranges, is less than 0.5%.

From the above discussion, control objectives are satisfactorily achieved by the fuzzy controller. However, a question remains as to whether the fuzzy controller performs better or offers something more than a classical controller. To answer this question, comparisons are made between the fuzzy controller and a PI controller[1] for several tests. Settling time, Integral of Absolute Error (IAE), and Root-Mean-Square (RMS) error criteria are used for the comparisons.

The first test compares the two controllers during start-up. Figure 5.4 shows coal feed rate for both fuzzy and PI control. Unlike coal feed rate for fuzzy control, feed rate for PI control was immediately set to the desired value because a logical stepping up of coal feed rate simply does not exist for conventional control. Also, an interesting difference between controllers is that the transition from LP gas preheating to coal feeding is handled by the fuzzy controller in a robust manner. The same fuzzy control law is used for both operating régimes whereas the PI controller needs to use gain scheduling to manage the transition (see Appendix E).

Figure 5.5 illustrates start-up temperature responses for fuzzy and PI control. The PI controller shows a larger drop in temperature after LP gas has been shut off than does the fuzzy controller. Each controller roughly gives the same overshoot, but temperature for the PI controller reaches a $\pm 10°$F band about the setpoint 2.9 minutes faster than the fuzzy controller. Also, the IAE criterion for PI control is 18.8% smaller than for fuzzy control. Settling times and IAE results for this test, as well as all following tests, are summarized in Table 5.1.

[1]A description of the PI controller appears in Appendix E.

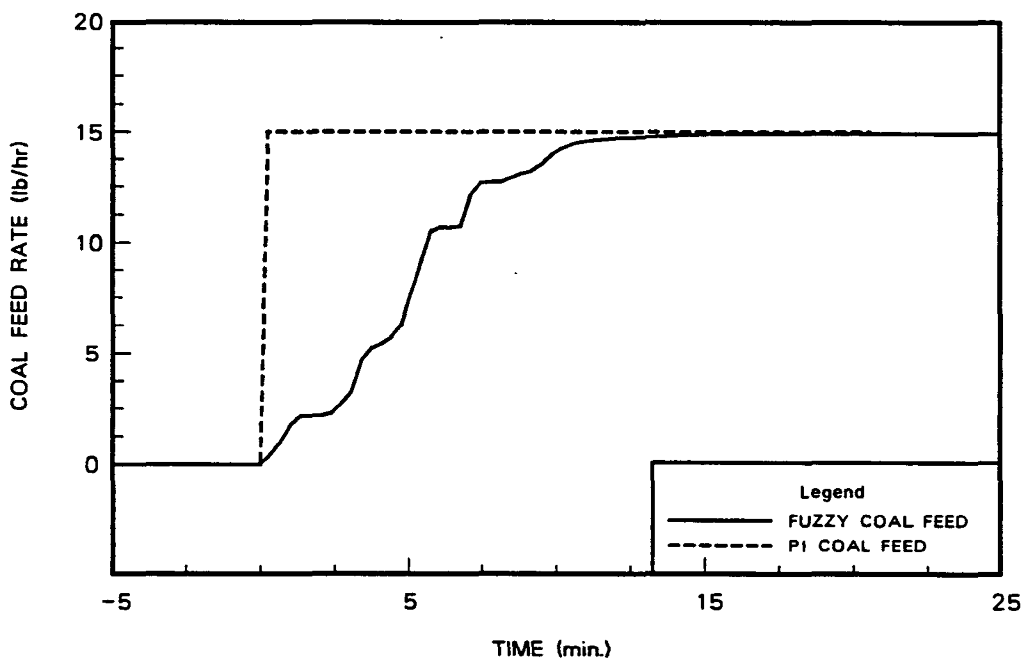

Figure 5.4: Coal Feed Rates for Fuzzy and PI Control During Start-up

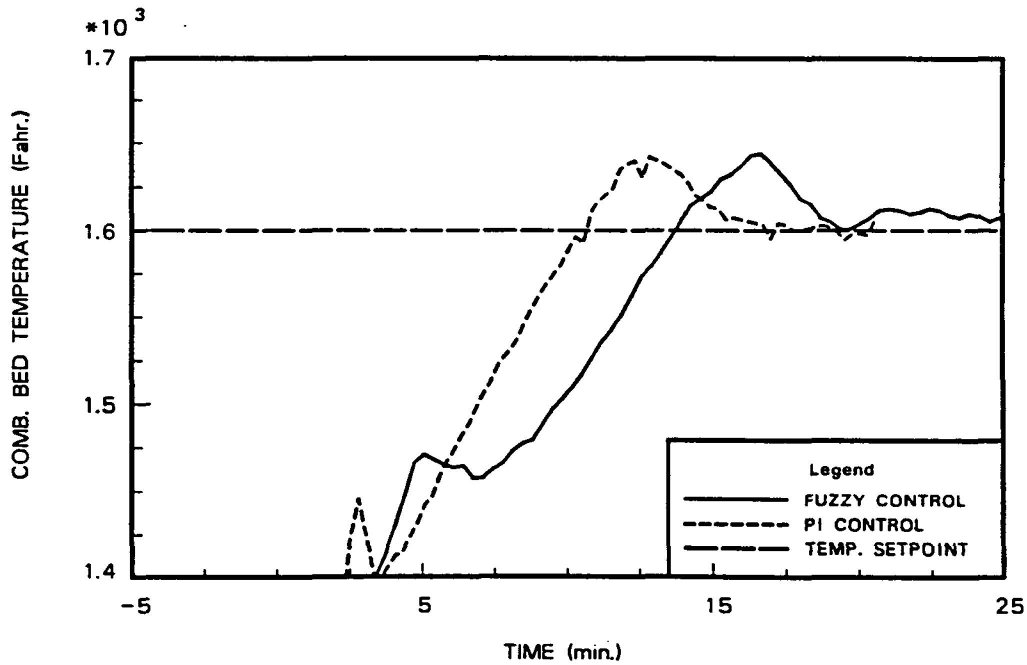

Figure 5.5: Temperature Responses for Fuzzy and PI Control During Start-up

Here, the PI controller's faster response is expected since a greater bed loading[2] is initially present due to immediately setting coal feed rate to the desired value. In this instance, the performance criteria are deceptive because fast heating can cause thermal stresses to crack the combustor's ceramic lining. Most large-scale operations require slow heating rates to reduce thermal stressing of their systems. Although not considered in this investigation, control loops for heating rate could have been added to both controllers.

Table 5.1: Comparison between Fuzzy and PI Control of Various Disturbance Rejections Using Settling Time and IAE Criteria

Disturbance		*Variable Affected*	$T_{settling}$[a]		IAE[b]	
			FUZZY	PI	FUZZY	PI
COAL FEED	Upward	Temp.	18.36	15.44	2177.6	1768.0
	Downward		7.49	6.36	86.4	52.7
	Upward	Oxy.	9.84	5.36	9.5	10.3
	Downward		3.40	1.27	2.4	1.4
PRI. AIR	Upward	Oxy.	2.73	2.30	5.0	2.6
	Downward		2.39	1.27	4.4	2.9
SEC. AIR	Upward	Temp.	6.81	3.84	222.1	77.9
	Downward		4.09	4.62	95.7	78.5

[a]For temperature responses, this is the time to enter a ±10°F band about 1600°F. For oxygen responses, this is the time to enter a ±0.75% band about 3.5% oxygen. In both cases, units are in minutes.

[b]Units are in °F·min. for temperature responses and %·min. for oxygen responses.

Figure 5.6 shows start-up oxygen responses for fuzzy and PI control. The PI controller exhibits a large, lengthy decrease in oxygen as coal feed rate is immediately set to the desired value. Oxygen response for fuzzy control has similar characteristics

[2]Bed loading refers to the amount of fuel present in the combustor at any time.

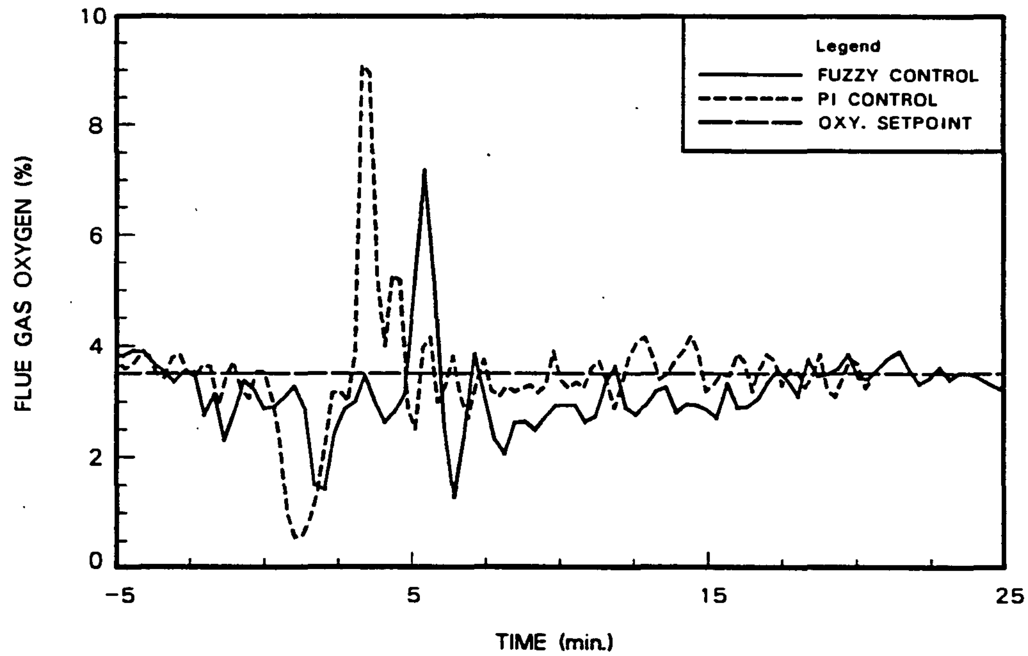

Figure 5.6: Oxygen Responses for Fuzzy and PI Control During Start-up

but not nearly as severe as those for PI control. After LP gas is turned off, the fuzzy controller has a smaller oxygen peak but also has a greater oxygen valley than does the PI controller. The PI controller causes oxygen to settle within a $\pm 0.75\%$ band about the setpoint 4.5 minutes faster but has 8.4% more IAE than the fuzzy controller.

During the start-up test, the PI controller used 17.5 scfm more primary air flow than did the fuzzy controller as shown in Figure 5.7. Unlike manual control, no maximum primary air flow limit is imposed during these tests. High air flow rates cause elutriation of small bed and fuel particles from the combustor. For this investigation, an oxygen criterion is used as a measure of combustor efficiency, but true efficiency can only be determined by evaluating the amount of unburned carbon in

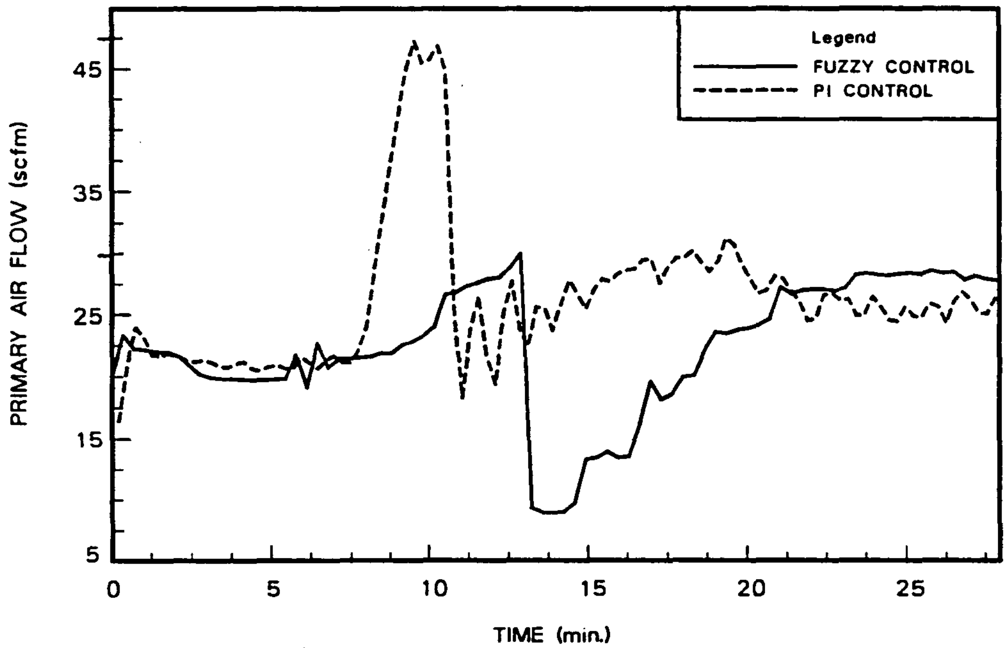

Figure 5.7: Primary Air Flows for Fuzzy and PI Control During Start-up

the fly ash. At the time these tests were conducted, no way existed to on-line quantify the amount of elutriated carbon. If this carbon could have been measured, the fuzzy controller would have appeared even more preferable to the PI controller than that indicated in Figure 5.7. These observations validate the "step-wise" approach taken to increment feed rate under fuzzy control.

The next test shows the controllers' responses for a 5 lb/hr reduction in coal feed rate (see Figure 5.8). Figure 5.9 depicts temperature responses for the two controllers. Both controllers demonstrate similar temperature drop for feed rate reduction. However, the PI controller exhibits a settling time 1.1 minutes faster than the fuzzy controller; also, IAE for the PI controller is 39% less than that for the fuzzy controller. Figure 5.10 shows flue gas oxygen responses for the feed rate reduction.

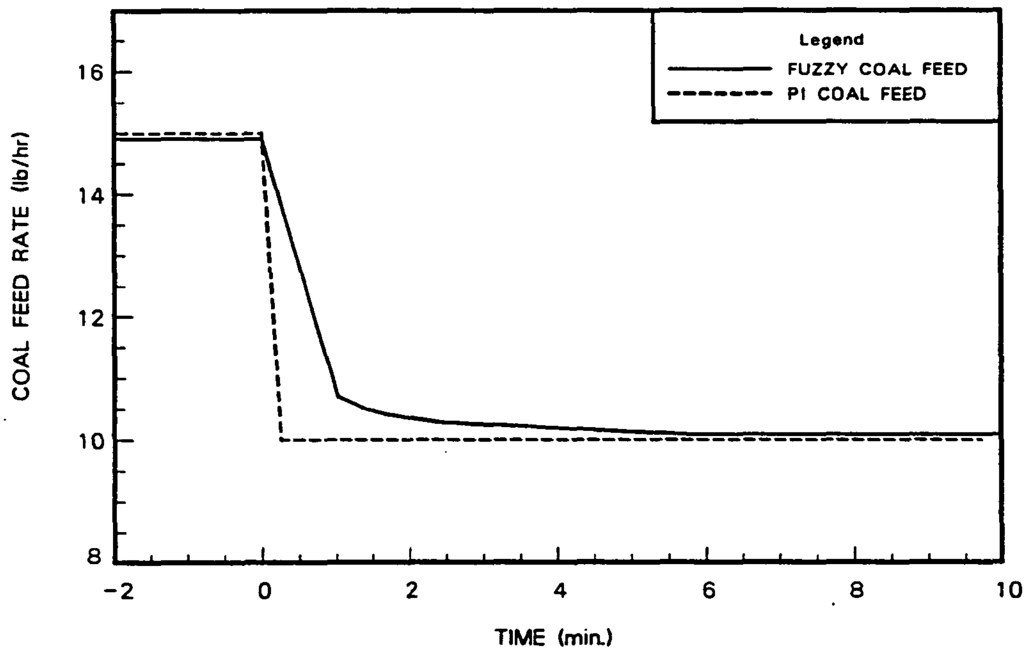

Figure 5.8: Coal Feed Rate Reductions for Fuzzy and PI Control

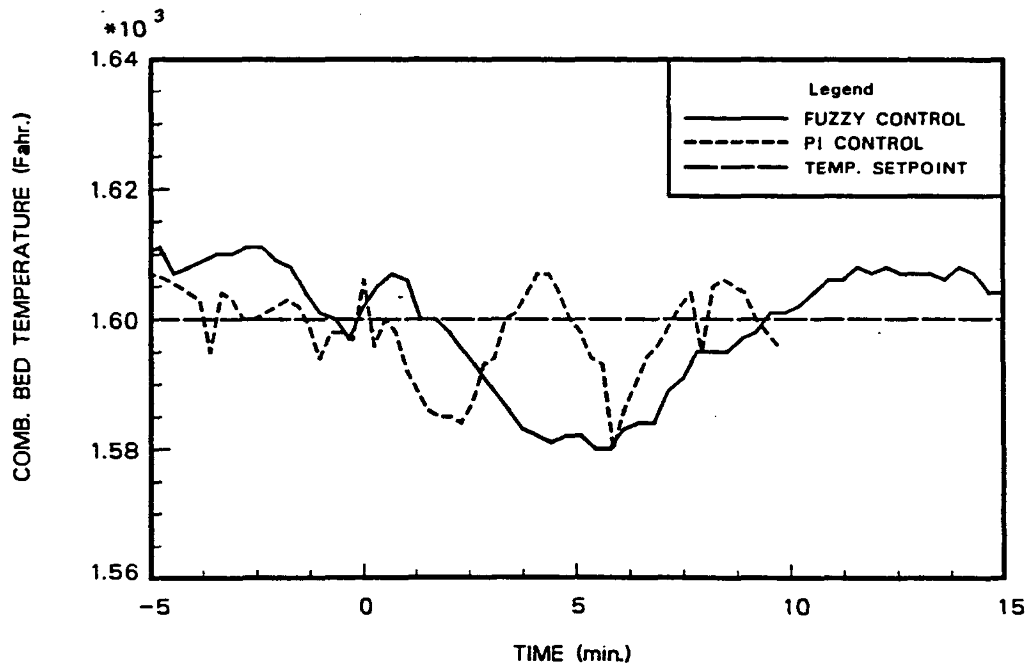

Figure 5.9: Temperature Responses for Fuzzy and PI Control for Feed Rate Reduction

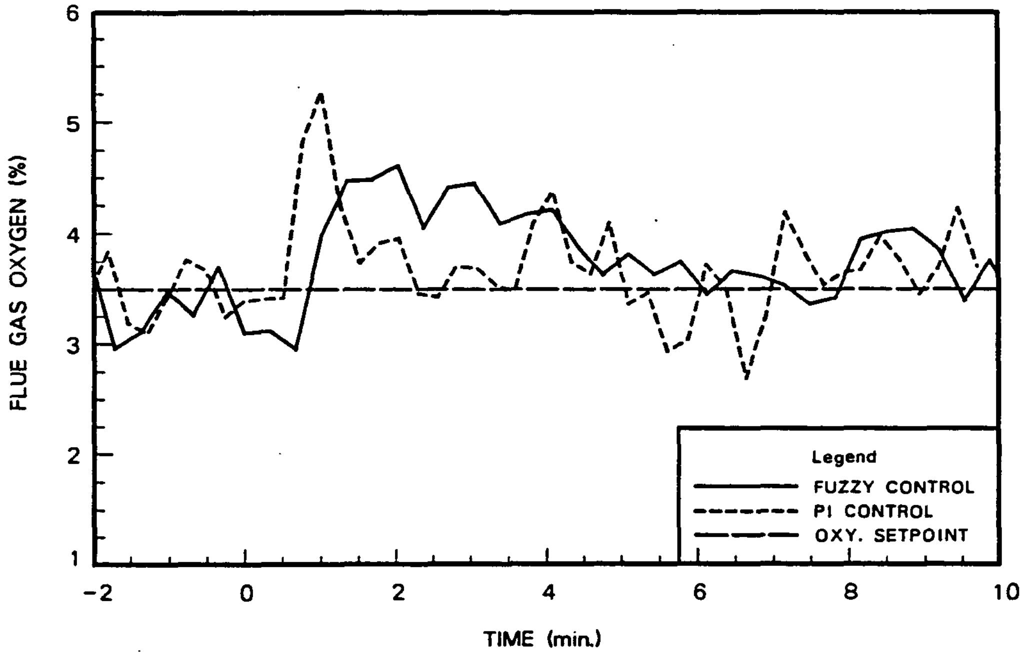

Figure 5.10: Oxygen Responses for Fuzzy and PI Control for Feed Rate Reduction

Although the PI controller has larger peak oxygen, it also has a settling time 2.1 minutes faster and 41.7% less IAE.

The next comparison is based upon the controllers' abilities to reject disturbances caused by manual adjustments to primary air. Combustor plumbing is such that the combustor can be operated either manually or automatically. With the manual control valves, auxiliary air flows can be adjusted to cause disturbances in both primary and secondary air flows. Figures 5.11 and 5.12 show oxygen responses of the controllers for a 15 scfm step increase and step decrease in primary air, respectively. For the step increase, the PI controller, having a slightly lower oxygen peak, realizes a settling time 0.43 minutes quicker and has 48% less IAE than the fuzzy controller. After each controller's response settled, auxiliary air flows were shut off forming step decreases in primary air flow. For these decreases, both controllers show similar oxygen drop. However, the PI controller shows a settling time 1.1 minutes quicker and an IAE 34.1% less than the fuzzy controller.

The next test shows the controllers reacting to secondary air flow disturbances. Figure 5.13 and 5.14 present temperature responses of the controllers for a 4 scfm step increase and step decrease in secondary air flow, respectively. For the step increase, the PI controller has about half the temperature drop shown for the fuzzy controller. Also, the PI controller exhibits a settling time 3.0 minutes quicker and 64.9% less IAE. For the step decrease, the fuzzy controller, having a slightly higher temperature rise, displays a settling time 0.53 minutes quicker, but its IAE is 21.9% larger than that for the PI controller.

Final comparisons of the controllers are made between RMS errors for temperature and oxygen regulation. Figures 5.15 and 5.16 compare steady-state temperatures and oxygen percentages, respectively. Each figure represents the longest available steady-state response for each controller. Also, Table 5.2 summarizes the numeric values of the errors. The fuzzy controller dominates this test by exhibiting less RMS steady-state error for both temperature and oxygen.

Table 5.2: Comparison between Fuzzy and PI Control of Steady-state RMS Error

Control of:	*RMS Error*	
	FUZZY	PI
Temperature (°F)	2.32	4.04
Oxygen (%)	0.194	0.300

As a final word, the PI controller's faster responses, other than heating rate, are

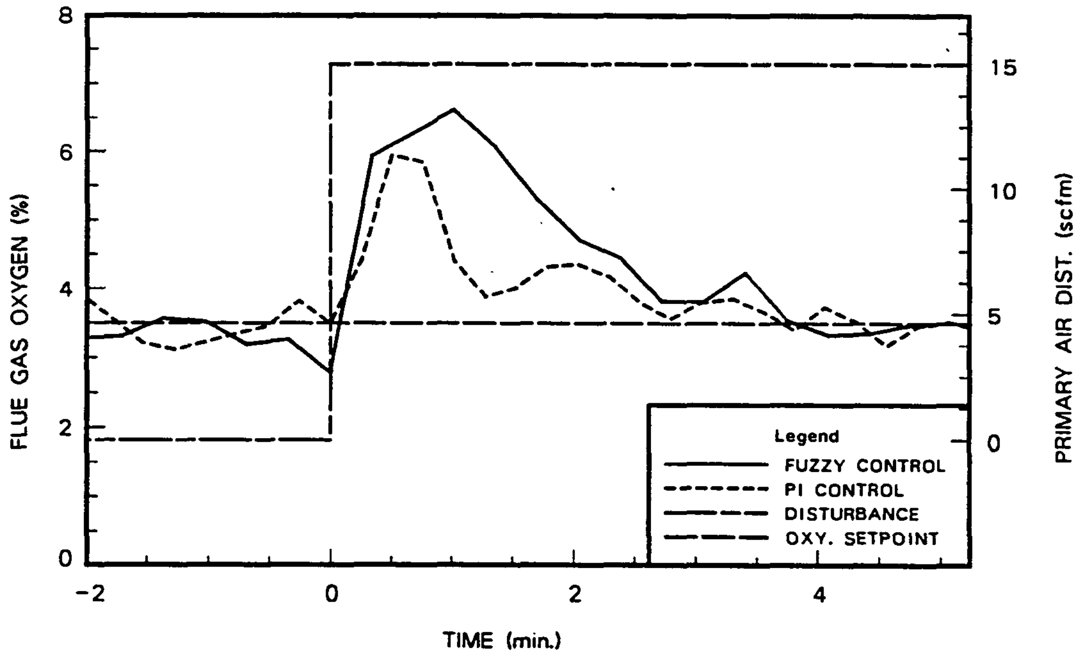

Figure 5.11: Oxygen Responses of Fuzzy and PI Control for Step-increased Primary Air

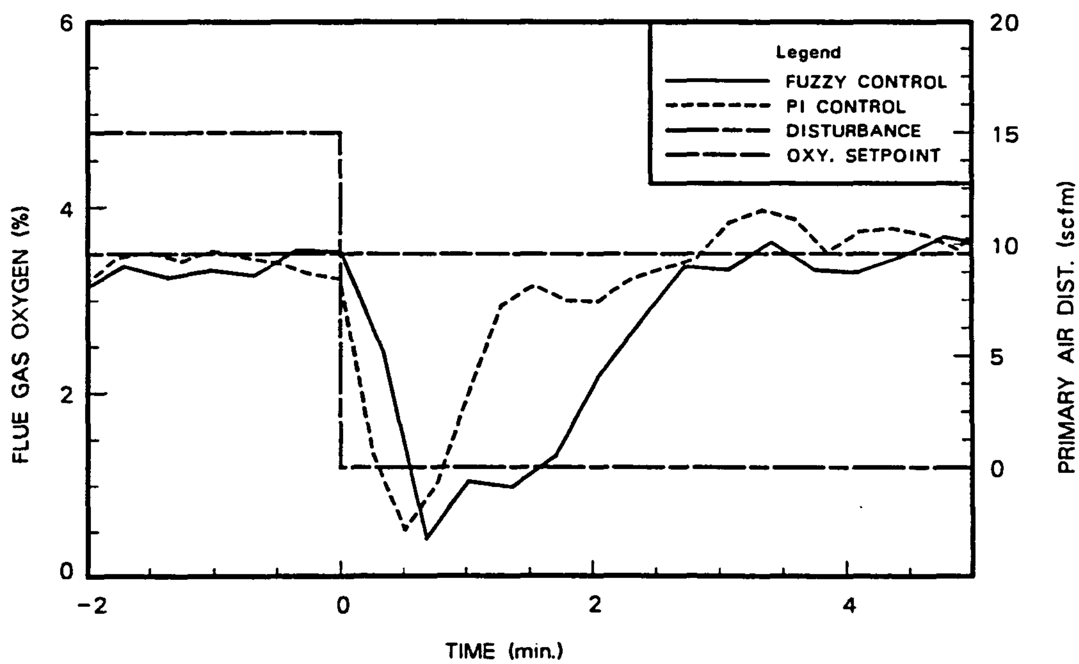

Figure 5.12: Oxygen Responses of Fuzzy and PI Control for Step-decreased Primary Air

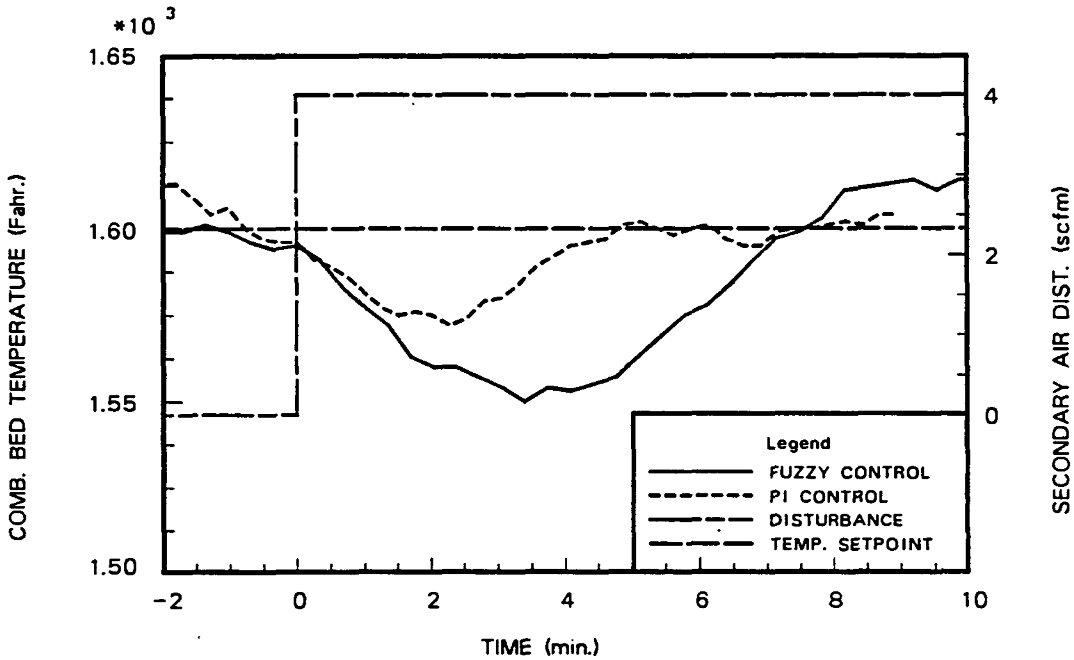

Figure 5.13: Temperature Responses of Fuzzy and PI Control for Step-increased Secondary Air

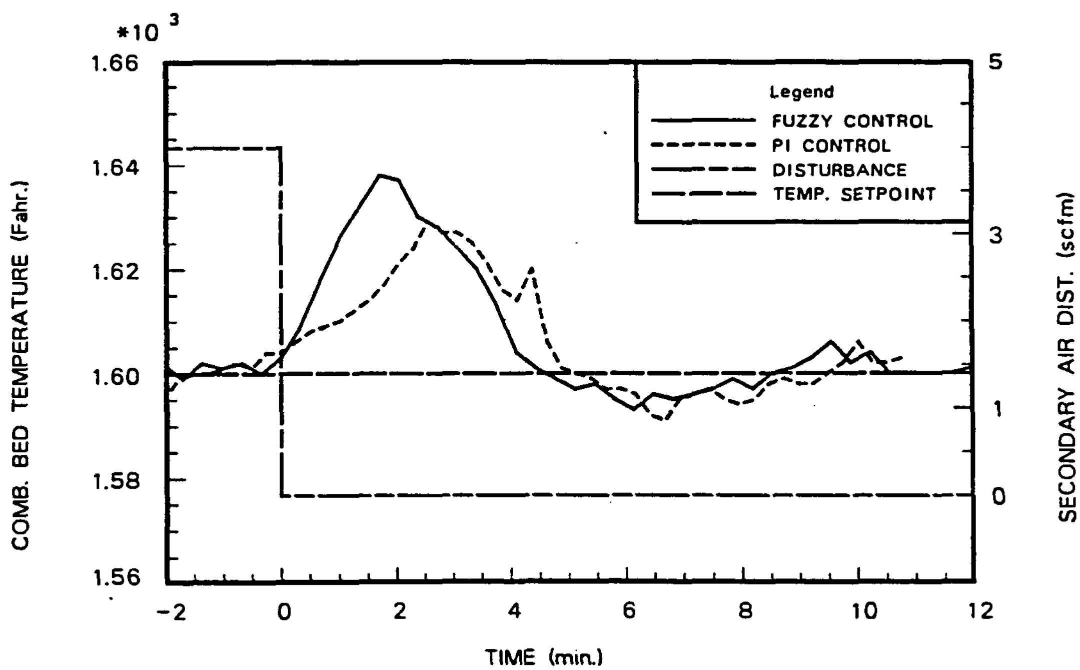

Figure 5.14: Temperature Responses of Fuzzy and PI Control for Step-decreased Secondary Air

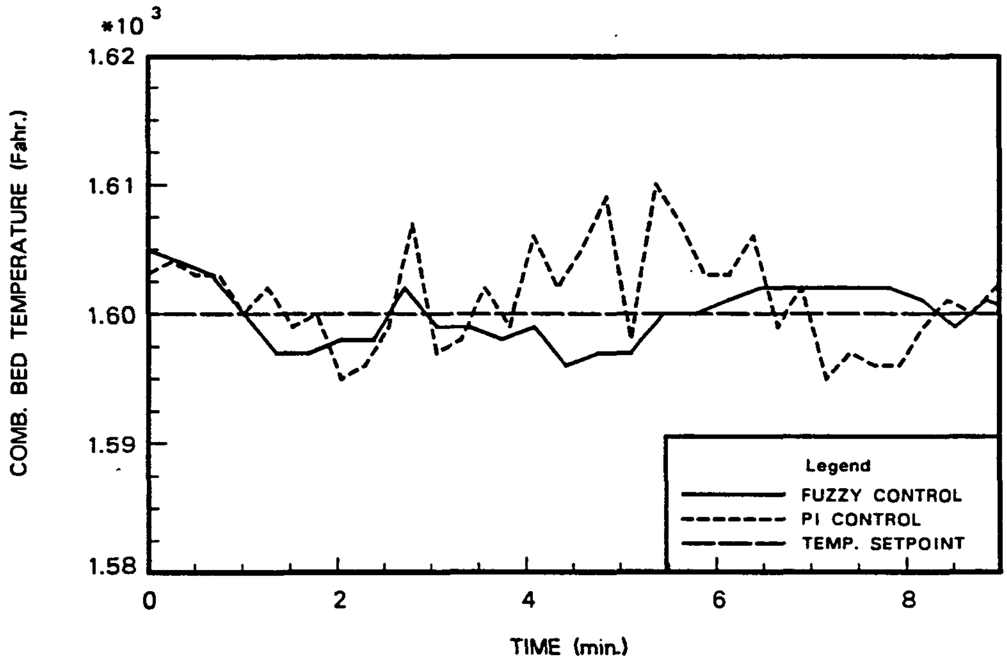

Figure 5.15: Steady-state Temperatures for Fuzzy and PI Control

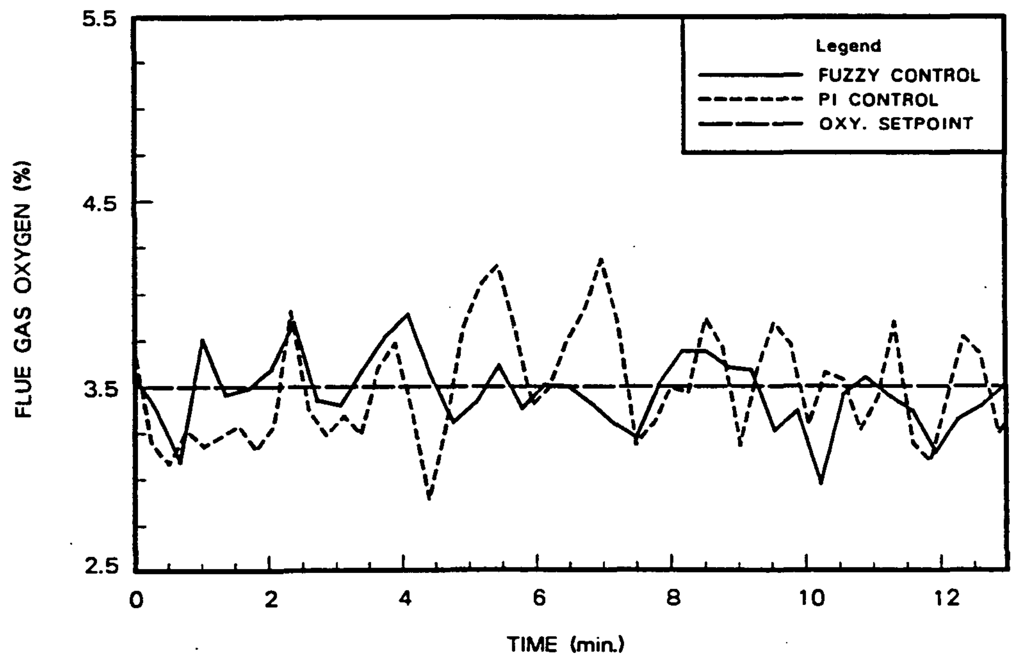

Figure 5.16: Steady-state Oxygen Percentages for Fuzzy and PI Control

attributed mainly to its faster sampling time of 0.6 seconds as opposed to 20 seconds for the fuzzy controller. The combustor under fuzzy control has 19.4 seconds more than the combustor under PI control to wander off course before control adjustments are made. Recall that the fuzzy controller's sampling time was selected/limited by human cognition of combustor responses. A scheme could be devised to project fuzzy controller actions designed with a "cognitive" time of 20 seconds back to quicker times by normalizing controller increments. That is, control actions could be divided by 20 and multiplied by a new sampling time to get projected incremental control actions. Such a controller should exhibit better performance than the nominal fuzzy controller.

6. CONCLUSIONS

The fuzzy controller satisfactorily achieved the desired objectives during start-up and steady-state operation. The PI controller, used for evaluating the fuzzy controller's performance, also gave desirable responses, usually faster than the fuzzy controller. The PI controller's faster response times are mainly attributed to the controller's faster sampling rate. Although the fuzzy controller's performance was good, it could be improved by projecting control actions to quicker sampling rates by normalizing nominal control actions.

Even with the lengthy 20-second sampling time, the fuzzy controller is preferred during start-up because of its much lower primary air flow requirement for good oxygen control. Although not explicitly addressed, the fuzzy controller also exhibits a robust nature during the transition between LP gas preheating and coal feeding whereas the PI controller needs to use gain scheduling to manage the transition. This characteristic of the fuzzy controller is desirable because future coal quality can be widely variable. The fuzzy controller's innate ability to manage large differences in fuels is better than the PI controller's design based on current fuel quality. As for steady-state operation, the PI controller is preferred because of its quicker rejection of disturbances.

The possibility of using a hybrid controller for FBCs is suggested — fuzzy control during start-up and PI control during normal operation. However, further tests with the fuzzy controller using a faster sampling time need to be conducted before trying this approach. A faster fuzzy controller may perform better than the PI controller in which case fuzzy control could be used exclusively. Also, since small-scale application of fuzzy logic proved useful, large-scale implementation should be examined. To this end, fuzzy logic could also be used to control combustor heating rate to reduce thermal stressing, and firing rate for changing power loads. For the latter case, fuzzy logic could be used to anticipate future power demands based on seasons, weather, time of day, etc. Future work should address these possibilities.

7. BIBLIOGRAPHY

[1] Merritt, P.C. "New Coal-Burning Systems Are on the Threshold." *Coal Age*, **91**(1986): 105-113.

[2] Green, L. " 'They're Off!' in the Circulating FBC Handicap." *Coal Mining*, **22**(1985): 49-52.

[3] McFarlane, R.C., Hoffman, T.W., Taylor, P.A. and MacGregor, J.F. "Control of Fluidized Bed Reactors. 1. Modeling, Simulation, and Single-Loop Control Studies." *Industrial Engineering: Chemical Process Design and Development*, **22**(1983): 22-31.

[4] Åström, K.J. and Wittenmark, B. *Computer-Controlled Systems: Theory and Design*. Prentice-Hall, Inc., Englewood Cliffs, N.J., 1984, 186-187.

[5] Zadeh, L.A. "Fuzzy Sets." *Information and Control*, **8**(1965): 338-353.

[6] Zadeh, L.A. "Outline of a New Approach to Analysis of Complex systems and Decision Processes." *IEEE Transactions on Systems, Man, and Cybernetics*, **SMC-3**(1973): 28-44.

[7] Mamdani, E.H. "Application of Fuzzy Algorithms for Control of Simple Dynamic Plant." *Proceedings of the IEE*, **121**(1974): 1585-1588.

[8] King, P.J. and Mamdani, E.H. "The Application of Fuzzy Control Systems to Industrial Processes." *Automatica*, **13**(1977): 235-242.

[9] Procyk, T.J. and Mamdani, E.H. "A Linguistic Self-Organizing Process Controller." *Automatica*, **15**(1979): 15-30.

[10] Kickert, W.J.M. and Van Nauta Lemke, H.R. "The Application of Fuzzy Set Theory to Control a Warm Water Process." *Automatica*, **12**(1976): 301-308.

[11] Tong, R.M. "A Control Engineering Review of Fuzzy Systems." *Automatica*, **13**(1977): 559-569.

[12] Tong, R.M., Beck, M.B. and Latten, A. "Fuzzy Control of the Activated Sludge Wastewater Treatment Process." *Automatica*, **16**(1980): 695-702.

[13] Holmblad, L.P. and Ostergaard, J.J. "Control of a Cement Kiln by Fuzzy Logic." In *Fuzzy Information and Decision Processes*, M.M. Gupta and E. Sanchez, eds. North-Holland, Amsterdam, 1982, 389-399.

[14] Larkin, L.I. "A Fuzzy Logic Controller For Aircraft Flight Control." In *Industrial Applications of Fuzzy Control*, M. Sugeno, ed. North-Holland, Amsterdam, 1985, 87-103.

[15] Tong, R.M. "Analysis of Fuzzy Control Algorithms Using the Relation Matrix." *International Journal of Man-Machine Studies*, **8**(1976): 679-686.

[16] Kunii, D. and Levenspiel, O. *Fluidization Engineering*. John Wiley & Sons, Inc., New York, 1969.

[17] Brown, R.C. and Foley, J.E. "A Method for Improving Load Turndown in Fluidized Bed Combustors." *Industrial And Engineering Chemistry Research*, **27**(1987): 24-30.

[18] Gibbs, B.M. and Hampartsoumian, E. "Limiting Air Pollution From Atmospheric Fluidised Bed Combustors." In *Fluidized Bed Boilers: Design and Application*, P. Basu, ed. Pergamon Press, Toronto, 1984, 131-154.

[19] Faulkenberry, L.M. *An Introduction to Operational Amplifiers with Linear IC Applications*. 2nd edition. John Wiley & Sons, New York, 1982, 7.

[20] *Linear Circuits Data Book*. Texas Instruments, Dallas, 1984, 3-226.

[21] Koffman, S.J., Fullmer, R.R. and Brown, R.C. "Fuzzy Logic Control of a Fluidized Bed Combustor." In *Proceedings of the 1989 American Control Conference*, Pittsburgh, June 1989, **3**, 2756-2758.

[22] Mano, M.M. *Digital Design*. Prentice-Hall, Inc., Englewood Cliffs, N.J., 1984, 72-83.

[23] Doebelin, E.O. *Control System Principles and Design*. John Wiley & Sons, New York, 1985, 485-486.

[24] Franklin, G.F. and Powell, J.D. *Digital Control of Dynamic Systems*. Addison-Wesley Publishing Co., Menlo Park, 1980, 97.

[25] Madsen, N.K. and Sincovec, R.F. The Numerical Methods of Lines for the Solution of Nonlinear Partial Differential Equations, Lawrence Livermore Laboratory, Rept. OCRL-75142, 1973.

[26] Van der Post, A.J., Bosgra, O.H., and Boelens, G. "Modelling the Dynamics of Fluidisation and Combustion in a Coal-Fired FBC." *Journal of Powder & Bulk Solids Technology*, **5**(1981): 32-37.

[27] Xavier, A.M. and Davidson, J.F. "Heat Transfer in Fluidized Beds: Convective Heat Transfer in Fluidized Beds." In *Fluidization*. $2^{\underline{nd}}$ edition. John Wiley & Sons, New York, 1985, 437-464.

8. APPENDIX A. COMPUTER CODES

8.1 QuickBASIC 4.0 Control Code

```
'*******************************************************************
'*                                                                 *
'*                 FUZZY LOGIC CONTROL PROGRAM                     *
'*                        (CALFLC.BAS)                             *
'*                                                                 *
'*                          5/10/89                                *
'*                                                                 *
'*******************************************************************

DIM Lt%(0 TO 1), DIGITAL%(3, 16)
COMMON SHARED Lt%(), DIGITAL%()
DECLARE SUB DAS8 (MODE%, BYVAL dummy%, flag%)

DECLARE SUB title ()
DECLARE SUB display ()
DECLARE SUB interpolation (tf!, voltage!(), board%, chan%)
```

```
DECLARE SUB inter ()

COMMON SHARED vk, svk, sik, Tk(), nk, cjc
COMMON SHARED FD(), FEED(), PD(), PFLOW(), SD(), SFLOW(), D()
COMMON SHARED CAFR, CAPA, CASA, ICAFR, ICAPA, ICASA

DIM GAIN(3), GASUM(6)
DIM SUM(3, 16), voltage(3, 16)
DIM temp(3, 16), orif(6, 2)
DIM Tk(400), Range(6), VOLT(6)
DIM scfm(2), T(3, 16), TSUM(3, 16), gasp(10)
DIM MFR(20, 20), MPA(20, 20), MSA(20, 20)
DIM SHARED PFLOW(100), SFLOW(100), PD(100), SD(100), FD(100)
DIM SHARED FEED(100), D(3)

'-----------------------------------------------------------------
'********************** DISPLAY TITLE BLOCK **********************
'-----------------------------------------------------------------

CALL title

'-----------------------------------------------------------------
'********************* INITIALIZE SYSTEM *************************
'-----------------------------------------------------------------
```

```
CLS
LOCATE 10, 20: PRINT "Initializing system ... please wait."

'-------------------------------------------------------------------
'********************* SYSTEM PARAMETERS ***************************
'-------------------------------------------------------------------

flag1 = 0          ' Controls printing of error messages
flag2 = 0          ' Controls coal feeding and allows
                   ' immediate reduction in primary air after
                   ' gas is shut off
flag3 = 0          ' Controls order of flow rate measurements
flag4 = 0          ' Allows proper control of red warning light
print.flag = 0

'-------------------------------------------------------------------
'*********************** KEY DEFINITIONS ***************************
'-------------------------------------------------------------------

KEY(1) ON: KEY(2) ON: KEY(3) ON: KEY(4) ON: KEY(5) ON: KEY(10) ON:
ON KEY(1) GOSUB key1
ON KEY(2) GOSUB key2
ON KEY(3) GOSUB key3
ON KEY(4) GOSUB key4
ON KEY(5) GOSUB key5
```

```
ON KEY(10) GOSUB key10

'------------------------------------------------------------------
'************************ DEFINE ARRAYS **************************
'------------------------------------------------------------------


'-------- Table lookup data for K type thermocouple -------------------
'Run this subroutine only in the initialization section of your program
'Number of points, voltage step interval (mV), starting voltage (mV)

DATA  309 ,  .2 , -6.6
READ nk, sik, svk

'Temperature at -6.6mv, -6.4mV, -6.2mV etc.

DATA -353.5,-249.3,-224.0,-207.6,-194.3,-182.8,-172.3,-162.8,-153.8,-145.4
DATA -137.3,-129.6,-122.3,-115.2,-108.3,-101.6, -95.1, -88.7, -82.5, -76.4
DATA  -70.4, -64.6, -58.8, -53.1, -47.5, -42.0, -36.6, -31.2, -25.9, -20.6
DATA  -15.4, -10.2,  -5.1,  -0.0,   5.0,  10.1,  15.1,  20.0,  25.0,  29.9
DATA   34.8,  39.7,  44.6,  49.5,  54.3,  59.1,  64.0,  68.8,  73.6,  78.4
DATA   83.2,  88.0,  92.9,  97.7, 102.5, 107.4, 112.2, 117.1, 122.0, 126.9
DATA  131.8, 136.7, 141.7, 146.6, 151.6, 156.5, 161.5, 166.5, 171.5, 176.5
DATA  181.6, 186.6, 191.6, 196.6, 201.6, 206.6, 211.6, 216.6, 221.5, 226.5
DATA  231.5, 236.4, 241.4, 246.3, 251.2, 256.1, 261.0, 265.9, 270.8, 275.6
```

```
DATA  280.5, 285.3, 290.2, 295.0, 299.8, 304.6, 309.4, 314.3, 319.1, 323.9
DATA  328.7, 333.4, 338.2, 343.0, 347.8, 352.6, 357.3, 362.1, 366.9, 371.6
DATA  376.4, 381.1, 385.9, 390.6, 395.4, 400.1, 404.8, 409.6, 414.3, 419.0
DATA  423.8, 428.5, 433.2, 437.9, 442.6, 447.3, 452.0, 456.8, 461.5, 466.2
DATA  470.9, 475.6, 480.3, 485.0, 489.7, 494.4, 499.1, 503.8, 508.5, 513.1
DATA  517.8, 522.5, 527.2, 531.9, 536.6, 541.3, 546.0, 550.7, 555.4, 560.0
DATA  564.7, 569.4, 574.1, 578.8, 583.5, 588.2, 592.9, 597.6, 602.3, 607.0
DATA  611.7, 616.4, 621.2, 625.9, 630.6, 635.3, 640.0, 644.8, 649.5, 654.2
DATA  658.9, 663.7, 668.4, 673.2, 677.9, 682.7, 687.4, 692.2, 696.9, 701.7
DATA  706.5, 711.3, 716.1, 720.8, 725.6, 730.4, 735.2, 740.0, 744.8, 749.7
DATA  754.5, 759.3, 764.1, 769.0, 773.8, 778.7, 783.5, 788.4, 793.3, 798.1
DATA  803.0, 807.9, 812.8, 817.7, 822.6, 827.5, 832.4, 837.3, 842.2, 847.2
DATA  852.1, 857.1, 862.0, 867.0, 872.0, 876.9, 881.9, 886.9, 891.9, 896.9
DATA  901.9, 906.9, 911.9, 916.9, 922.0, 927.0, 932.0, 937.1, 942.2, 947.2
DATA  952.3, 957.4, 962.5, 967.6, 972.7, 977.8, 982.9, 988.0, 993.1, 998.2
DATA 1003.4,1008.5,1013.7,1018.8,1024.0,1029.2,1034.4,1039.6,1044.8,1050.0
DATA 1055.2,1060.4,1065.6,1070.8,1076.1,1081.3,1086.6,1091.9,1097.2,1102.4
DATA 1107.7,1113.0,1118.3,1123.7,1129.0,1134.3,1139.7,1145.0,1150.4,1155.8
DATA 1161.2,1166.6,1172.0,1177.4,1182.9,1188.3,1193.8,1199.2,1204.7,1210.2
DATA 1215.7,1221.2,1226.8,1232.3,1237.9,1243.5,1249.1,1254.7,1260.3,1265.9
DATA 1271.6,1277.3,1282.9,1288.6,1294.3,1300.1,1305.8,1311.5,1317.3,1323.1
DATA 1328.9,1334.7,1340.5,1346.4,1352.2,1358.1,1363.9,1369.8,1375.7

        FOR I = 0 TO nk - 1
              READ Tk(I)
```

```
        NEXT I

'------------------------------------------------------------------

'***************** ASSIGNMENT OF ORIFICE CONSTANTS ****************

'------------------------------------------------------------------

DATA 2.3426,0.51324,1.8196,0.50523,1.2647,0.51488,1.3359,0.5005

DATA 0.8500,0.53392,0.34717,0.49591

        FOR III = 1 TO 6

                FOR JJJ = 1 TO 2

                        READ orif(III, JJJ)

                NEXT JJJ

        NEXT III

' SET RANGES FOR GAS ANALYZERS

Range(1) = 25: VOLT(1) = 5                                  ' OXYGEN

Range(2) = 1.2: VOLT(2) = 5                                 ' CARBON MONOXIDE

Range(3) = 30: VOLT(3) = 5                                  ' CARBON DIOXIDE

Range(4) = 2000: VOLT(4) = 1                                ' SULFUR DIOXIDE

Range(5) = 1000: VOLT(5) = 1                                ' NOx

'SET THE GAINS FOR THE THREE EXP-16'S
```

```
GAIN(1) = 50

GAIN(2) = 1000

GAIN(3) = 50


'FIND COLD JUNCTION TEMPERATURE(24.4 MV/DEG C)


adr% = 768

MODE% = 0: flag% = 0

CALL DAS8(MODE%, VARPTR(adr%), flag%)

IF flag% <> 0 THEN LOCATE 1, 5: PRINT "ERROR 1.0: DAS-8 INSTALLATION"


MODE% = 1: Lt%(0) = 0: Lt%(1) = 0

CALL DAS8(MODE%, VARPTR(Lt%(0)), flag%)

IF flag% <> 0 THEN LOCATE 1, 5: PRINT " ERROR Main 1.2: CJCT"

CJSUM = 0

RRR = 10

        FOR J = 1 TO RRR

                MODE% = 4: NUM% = 0

                CALL DAS8(MODE%, VARPTR(NUM%), flag%)

                IF flag% <> 0 THEN LOCATE 1, 5: PRINT " Error Main 3.1: CJ "

                CJSUM = CJSUM + NUM%

        NEXT J

cjc = (CJSUM * 5!) / (2047! * .0244 * RRR)


'-------------------------------------------------------------------
```

```
'******************* READ IN CALIBRATION DATA ***********************
'--------------------------------------------------------------------

OPEN "PRICAL.DAT" FOR INPUT AS #1
OPEN "SECCAL.DAT" FOR INPUT AS #2
OPEN "AUGCAL.DAT" FOR INPUT AS #3

FOR X = 1 TO 81
INPUT #1, PD(X), PFLOW(X)
NEXT X

FOR X = 1 TO 81
INPUT #2, SD(X), SFLOW(X)
NEXT X

FOR X = 1 TO 20
INPUT #3, FD(X), FEED(X)
NEXT X

CLOSE #1, #2, #3

'--------------------------------------------------------------------
'**************** READ IN FUZZY LOGIC LOOK-UP TABLE *****************
'--------------------------------------------------------------------
```

```
OPEN "FEED.FLC" FOR INPUT AS #1
OPEN "PRI.FLC" FOR INPUT AS #2
OPEN "SEC.FLC" FOR INPUT AS #3

FOR J = 0 TO 20
FOR I = 0 TO 20
INPUT #1, MFR(I, J)
INPUT #2, MPA(I, J)
INPUT #3, MSA(I, J)
NEXT I: NEXT J

CLOSE #1, #2, #3

'--------------------------------------------------------------------
'****************** ENTER PARAMETERS FROM KEYBOARD ******************
'--------------------------------------------------------------------

CLS
LOCATE 3, 15: PRINT "*** PARAMETER INITIALIZATION ***"
LOCATE 6, 5
INPUT "Enter Line Gage Pressure (psig): ", Gage
pressure = Gage + 14.7

LOCATE 8, 5
INPUT "Enter control delay (sec): ", delay
```

```
LOCATE 10, 5
INPUT "Enter scaling factor for primary air: ", KPA

LOCATE 12, 5
INPUT "Enter scaling factor for secondary air: ", KSA

LOCATE 14, 5
INPUT "Enter scaling factor for coal feed rate: ", KFR

again2:
LOCATE 16, 5
INPUT "Enter coal feed rate (lb/hr): ", frate
LOCATE 16, 5: PRINT "                                                  "
IF frate < 5 OR frate > 20 THEN GOTO again2:

'---------------------------------------------------------------------
'******************** OUTPUT FILE INITIALIZATION ********************
'---------------------------------------------------------------------

import:
CLS
LOCATE 3, 15: PRINT "*** OUTPUT FILE INITIALIZATION ***"
LOCATE 5, 5: PRINT "NOTE:FLC will create a Lotus 1-2-3"
LOCATE 6, 5: PRINT "     IMPORT format file (PRN extension)"
```

```
LOCATE 10, 5: PRINT "Enter Output File Name:                    .PRN"
LOCATE 10, 33: INPUT "", dfile$

LOCATE 19, 5: PRINT "Edit? (y/[n])"

DO
ans$ = INKEY$
LOOP UNTIL ans$ <> ""

IF ans$ = "y" OR ans$ = "Y" THEN GOTO import:

OPEN dfile$ + ".prn" FOR OUTPUT AS #1
WRITE #1, "Elapse"
WRITE #1, "Delay"
WRITE #1, "Selected Temp"
WRITE #1, "Temp Change"
WRITE #1, "Temp Setpoint"
WRITE #1, "Annular Bed 1"
WRITE #1, "Primary Air"
WRITE #1, "Secondary Air"
WRITE #1, "Oxygen"
WRITE #1, "Oxy Change"
WRITE #1, "Oxy Setpoint"
WRITE #1, "Pri Air Action"
WRITE #1, "Sec Air Action"
```

```
WRITE #1, "Feed Rate"
WRITE #1, "KPA"
WRITE #1, "KSA"
WRITE #1, "KFR"
Dcount = 0
settemp = 1600
setoxy = 3.5

'-------------------------------------------------------------------
'**************** DETERMINE PRESSURE TRANSDUCER OFFSET **************
'-------------------------------------------------------------------
CLS
LOCATE 10, 5: PRINT "Please wait ... calculating pressure transducer offset."

OUT &H305, 2 ^ 0

time1 = TIMER            ' Allow pressure transients to be damped out
DO:
time2 = TIMER
LOOP UNTIL time2 - time1 > 10!

zoff = 0!
GOSUB press:
zoff = inches.air
CLS
```

```
LOCATE 10, 5
PRINT "Measured pressure transducer offset (in. of H2O): "; USING "##.##"; zoff

time1 = TIMER            ' Delay
DO:
time2 = TIMER
LOOP UNTIL time2 > time1 + 5!

'------------------------------------------------------------------------
'********************** WARM BED WITH LP GAS ***************************
'------------------------------------------------------------------------

top:
CSUM = 2 ^ 0             ' Switch pressure transducer to primary air
OUT &H305, CSUM          ' (flag3 = 0)
FOR N = 0 TO 3
OUT &H350 + 2 * N, 0
OUT &H350 + 1 + 2 * N, 0
NEXT N

CLS
LOCATE 5, 24: PRINT "*** BED WARMING ROUTINE ***"

LOCATE 10, 7: PRINT "In addition to reading and understanding the manual
                     control"
```

```
LOCATE 11, 7: PRINT "instructions, the upcoming instructions must be strictly
                     followed."

LOCATE 17, 25: PRINT "Press space bar to continue."

DO: LOOP UNTIL INKEY$ = " "

CLS
LOCATE 5, 24: PRINT "*** BED WARMING ROUTINE ***"

LOCATE 10, 7: PRINT "1.)   Prepare gas analyzers and sampling line."

LOCATE 20, 15: PRINT "AFTER completing step 1, press space bar to continue."

DO: LOOP UNTIL INKEY$ = " "

CLS
LOCATE 5, 24: PRINT "*** BED WARMING ROUTINE ***"

LOCATE 10, 7: PRINT "2.)   Turn on exhaust fan and properly route exhaust"
LOCATE 11, 7: PRINT "      from the blue combustor to the fan."

LOCATE 20, 15: PRINT "AFTER completing step 2, press space bar to continue."

DO: LOOP UNTIL INKEY$ = " "
```

```
CLS

LOCATE 5, 24: PRINT "*** BED WARMING ROUTINE ***"

LOCATE 10, 7: PRINT "3.)   Turn on main water valve (below steam and air ports"
LOCATE 11, 7: PRINT "      on work table) and turn on water to heat exchanger"
LOCATE 12, 7: PRINT "      on exhaust pipe."

LOCATE 20, 15: PRINT "AFTER completing step 3, press space bar to continue."

DO: LOOP UNTIL INKEY$ = " "

CLS

LOCATE 5, 24: PRINT "*** BED WARMING ROUTINE ***"

LOCATE 10, 7: PRINT "4.)   Turn on power and air to pnuematic valves."
LOCATE 11, 7: PRINT "      Turn on power to auger feeder and set control"
LOCATE 12, 7: PRINT "      switch to automatic."

LOCATE 20, 15: PRINT "AFTER completing step 4, press space bar to continue."

DO: LOOP UNTIL INKEY$ = " "

CLS

LOCATE 5, 24: PRINT "*** BED WARMING ROUTINE ***"
```

```
' Turn on water, primary air, and secondary air.

D(1) = 2500             ' water (static variable)
CAFR = 0                ' fuel feed rate (dynamic variable)
CAPA = 15               ' primary air flow rate (dynamic variable)
CASA = 5                ' secondary air flow rate (dynamic variable)

CALL inter              ' interpolate and output actions

LOCATE 10, 7: PRINT "5.)  At this point, the follow should have occurred:"
LOCATE 11, 7: PRINT "        a.) the cooling water has come on,"
LOCATE 12, 7: PRINT "        b.) the combustion bed has been fluidized, and"
LOCATE 13, 7: PRINT "        c.) the inner bed has received an air flow."
LOCATE 15, 7: PRINT "     Make sure that the above have occurred."

LOCATE 20, 15: PRINT "AFTER completing step 5, press space bar to continue."

DO: LOOP UNTIL INKEY$ = " "

CLS
LOCATE 5, 24: PRINT "*** BED WARMING ROUTINE ***"

LOCATE 10, 7: PRINT "6.)  Plug all necessary electric cords into their proper"
LOCATE 11, 7: PRINT "     location on the computer-controlled outlet board."
```

```
LOCATE 20, 15: PRINT "AFTER completing step 6, press space bar to continue."

DO:
LOCATE 14, 7: PRINT "BEWARE: BY COMPLETING THIS STEP, THE SPARKER OUTLET"
LOCATE 15, 7: PRINT "         WILL BE ENGERIZED! "
FOR X = 1 TO 5000: NEXT X
LOCATE 14, 7: PRINT "                                                    "
LOCATE 15, 7: PRINT "                           "
FOR X = 1 TO 2000: NEXT X
LOOP UNTIL INKEY$ = " "

CSUM = CSUM + 2 ^ 7
OUT &H305, CSUM           ' turn on sparker outlet

CLS
LOCATE 5, 24: PRINT "*** BED WARMING ROUTINE ***"

LOCATE 10, 7: PRINT "7.)   Turn on the switch for electrode sparker and"
LOCATE 11, 7: PRINT "      make sure that the sparker is working properly."

LOCATE 20, 15: PRINT "AFTER completing step 7, press space bar to continue."

DO: LOOP UNTIL INKEY$ = " "
```

```
CLS
LOCATE 5, 24: PRINT "*** BED WARMING ROUTINE ***"

LOCATE 10, 7: PRINT "8.)   Turn on the valve on top of the LP gas cylinder"
LOCATE 11, 7: PRINT "      and set the gas directional valve for the blue
                       combustor."

LOCATE 20, 15: PRINT "AFTER completing step 8, press space bar to continue."

DO: LOOP UNTIL INKEY$ = " "

flag4 = 1
CSUM = CSUM + 2 ^ 5      ' Turn on red warning light
OUT &H305, CSUM

CLS
LOCATE 5, 24: PRINT "*** BED WARMING ROUTINE ***"

LOCATE 10, 7: PRINT "This is your LAST chance to be sure that all is operating"
LOCATE 11, 7: PRINT "properly to this point."

LOCATE 13, 7: PRINT "To initiate ignition, press I."
LOCATE 14, 7: PRINT "To review previous steps, press R."
LOCATE 15, 7: PRINT "To quit, press F[10]."
```

```
again:
LOCATE 17, 7: PRINT "BEWARE: BY TYPING   < I >   THE LP GAS WILL COME ON!"
FOR X = 1 TO 5000: NEXT X
X$ = INKEY$
IF X$ = "I" OR X$ = "i" GOTO start:
IF X$ = "R" OR X$ = "r" GOTO top:
LOCATE 17, 7: PRINT "                                                    "
FOR X = 1 TO 2000: NEXT X
X$ = INKEY$
IF X$ = "I" OR X$ = "i" GOTO start:
IF X$ = "R" OR X$ = "r" GOTO top:
GOTO again:


start:
CLS
LOCATE 5, 24: PRINT "*** BED WARMING ROUTINE ***"
LOCATE 10, 30: PRINT "Please wait."
LOCATE 15, 12: PRINT "Initial temperature= "
LOCATE 16, 12: PRINT "Current temperature= "
LOCATE 20, 17: PRINT "PRESS F[10] FOR EMERGENCY PROGRAM TERMINATION."


GOSUB Thermocouple:             ' Get initial temperature
temp1 = temp(1, 2)
LOCATE 15, 33: PRINT USING "####.#"; temp1
```

```
CSUM = CSUM + 2 ^ 6        ' Turn on LP gas
OUT &H305, CSUM

time1 = TIMER              ' Delay
DO:
time2 = TIMER
LOOP UNTIL time2 > time1 + 5!

CAPA = 25                                 ' Fluidize combustion bed
CALL inter

time1 = TIMER              ' time delay
DO:
GOSUB Thermocouple:
temp2 = temp(1, 2)
LOCATE 16, 33: PRINT USING "####.#"; temp2
time2 = TIMER
LOOP UNTIL time2 > time1 + 20!

CLS

IF temp2 < temp1 + 10! THEN
        CSUM = CSUM - 2 ^ 6
        OUT &H305, CSUM
        LOCATE 5, 24: PRINT "*** BED WARMING ROUTINE ***"
```

```
        LOCATE 10, 23: PRINT "UNSUCCESSFUL LP GAS IGNITION!"
        LOCATE 11, 24: PRINT "START-UP PROCEDURE ABORTED!"
        LOCATE 18, 12: PRINT "Please check all electrical and mechanical
                              connections."
        LOCATE 19, 12: PRINT "To try again, press any key."
        DO: LOOP UNTIL INKEY$ <> ""
        GOTO top:
END IF

flag4 = 0
CSUM = CSUM + 2 ^ 3 - 2 ^ 5                ' Turn on sand purger and off
OUT &H305, CSUM                            ' red warning light
CAPA = 20                                  ' Fluidize combustion bed
CALL inter

'----------------------------------------------------------------------
'********************* WRITE HEADING TO SCREEN **********************
'----------------------------------------------------------------------

CALL display

'----------------------------------------------------------------------
'************************** CONTROL LOOP **************************
'----------------------------------------------------------------------
```

```
time.start = TIMER              ' Get starting time
timesub1 = 0
print.flag = 1
oxy.old = gasp(1)

100 :

Time.now1 = TIMER
TEMPS.OLD = TEMPS

'--------------------------------------------------------------------
'******** GET BOTH PRIMARY AND SECONDARY AIR FLOW RATES *************
'--------------------------------------------------------------------

IF flag3 = 0 THEN
        ornum = 1                         ' Primary Air
        GOSUB press
        scfm(1) = flow

        ornum = 3                         ' Secondary Air
        CSUM = CSUM + 2 ^ 1 - 2 ^ 0       ' Turn relay #0 off and relay #1 on
        OUT &H305, CSUM

        Tref = TIMER                      ' Let pressure transducer stabilize
        DO
```

```
        Tcomp = TIMER
        LOOP UNTIL Tcomp - Tref > 2

        GOSUB press
        scfm(2) = flow
        flag3 = 1
        GOTO place:
END IF

IF flag3 = 1 THEN
        ornum = 3                          ' Secondary Air
        GOSUB press
        scfm(2) = flow

        ornum = 1                          ' Primary Air
        CSUM = CSUM + 2 ^ 0 - 2 ^ 1        ' Turn relay #0 on and relay #1 off
        OUT &H305, CSUM

        Tref = TIMER                       ' Let pressure transducer stabilize
        DO
        Tcomp = TIMER
        LOOP UNTIL Tcomp - Tref > 2

        GOSUB press
        scfm(1) = flow
```

```
        flag3 = 0
END IF

' Print air flow data

place:
LOCATE 15, 67: PRINT USING "##.##"; scfm(1)
LOCATE 16, 67: PRINT USING "##.##"; scfm(2)

repeat:

'---------------------------------------------------------------------
'************ GET BED TEMPERATURE AND GAS CONCENTRATIONS ************
'---------------------------------------------------------------------

GOSUB Thermocouple

' Print temperature data

LOCATE 7, 21: PRINT USING "####.#"; temp(1, 1)
LOCATE 7, 36: PRINT USING "####.#"; temp(1, 2)
LOCATE 7, 51: PRINT USING "####.#"; temp(1, 3)
LOCATE 7, 71: PRINT USING "#"; PROBE
LOCATE 8, 21: PRINT USING "####.#"; temp(1, 4)
```

```
' Print gas data

LOCATE 10, 12: PRINT USING " ##.##"; gasp(1)
LOCATE 3, 67: PRINT USING "###.##"; delay

' Continuously update temperature and gas data

Time.now2 = TIMER
Elapse = (Time.now2 - time.start) / 60
LOCATE 3, 11: PRINT USING "###.##"; Elapse
IF Time.now2 < timesub1 + delay - 5! THEN oxy.old = gasp(1)
IF Time.now2 < timesub1 + delay THEN GOTO repeat:

timesub1 = TIMER

'--------------------------------------------------------------------
'********************** REGULAR CONTROL LOGIC ***********************
'--------------------------------------------------------------------

IF TEMPS > 1200 AND flag2 = 0 THEN flag2 = 1        ' Start coal feeding
IF flag2 = 0 THEN SUBCAPA = CAPA                    ' Remember flow rate for
                                                    ' gas feeding

IF TEMPS > 1450 AND CAFR > 4 AND flag1 = 0 THEN     ' Turn off gas, sparker, and
                                                    ' electrode purger
```

```
        CSUM = CSUM - 2 ^ 3 - 2 ^ 6 - 2 ^ 7
        OUT &H305, CSUM
        flag2 = 2
        flag1 = 1
END IF

IF flag1 = 1 AND TEMPS < 1400 AND flag4 = 0 THEN
        LOCATE 19, 8: PRINT "TEMPERATURE IS LOW."
        flag4 = 1
        CSUM = CSUM + 2 ^ 5     ' Turn on red warning light
        OUT &H305, CSUM
END IF

IF flag1 = 1 AND TEMPS < 1400 AND flag4 = 1 THEN
        LOCATE 19, 8: PRINT "TEMPERATURE IS LOW."
END IF

IF TEMPS > 1700 AND flag4 = 0 THEN
        LOCATE 19, 8: PRINT "TEMPERATURE IS HIGH."
        flag4 = 1
        CSUM = CSUM + 2 ^ 5     ' Turn on red warning light
        OUT &H305, CSUM
END IF

IF TEMPS > 1700 AND flag4 = 1 THEN
```

```
        LOCATE 19, 8: PRINT "TEMPERATURE IS HIGH."
END IF


IF gasp(1) < 1! AND flag4 = 0 THEN
        LOCATE 20, 8: PRINT "FLUE GAS OXYGEN IS LOW."
        flag4 = 1
        CSUM = CSUM + 2 ^ 5      ' Turn on red warning light
        OUT &H305, CSUM
END IF


IF gasp(1) < 1! AND flag4 = 1 THEN
        LOCATE 20, 8: PRINT "FLUE GAS OXYGEN IS LOW."
END IF


IF gasp(1) > 6! AND flag4 = 0 THEN
        LOCATE 20, 8: PRINT "FLUE GAS OXYGEN IS HIGH."
        flag4 = 1
        CSUM = CSUM + 2 ^ 5      ' Turn on red warning light
        OUT &H305, CSUM
END IF


IF gasp(1) > 6! AND flag4 = 1 THEN
        LOCATE 20, 8: PRINT "FLUE GAS OXYGEN IS HIGH."
END IF
```

```
'-------------------------------------------------------------------
'*********************** FUZZY CONTROL LOGIC ***********************
'-------------------------------------------------------------------

' Determine indices to reference control actions from control matrices.

TI = CINT(20! * (TEMPS - 1525!) / 150!)
IF TI < 0 THEN TI = 0
IF TI > 20 THEN TI = 20

CTEMPS = TEMPS - TEMPS.OLD
CTI = CINT(20! * (CTEMPS + 15!) / 30!)
IF CTI < 0 THEN CTI = 0
IF CTI > 20 THEN CTI = 20

OI = CINT(20! * (gasp(1) - 2!) / 3!)
IF OI < 0 THEN OI = 0
IF OI > 20 THEN OI = 20

coxy = gasp(1) - oxy.old
COI = CINT(20! * (coxy + .75) / 1.5)
IF COI < 0 THEN COI = 0
IF COI > 20 THEN COI = 20

FRI = CINT(20! * (frate - CAFR + 2!) / 4!)
```

```
IF FRI < 0 THEN FRI = 0

IF FRI > 20 THEN FRI = 20

IF flag2 = 0 THEN FRI = 10


' Read incremental control actions from control matrices.


IF flag2 = 0 THEN ICAFR = 0

IF flag2 <> 0 THEN ICAFR = KFR * (MFR(OI, FRI) - 10!) / 10!


IF flag2 = 0 THEN
        ICASA = 0
ELSE
        ICASA = KSA * (MSA(TI, CTI) - 10!) / 10!
END IF


IF flag2 <> 2 THEN ICAPA = KPA * (MPA(OI, COI) - 10!) / 10!
IF flag2 = 2 THEN
        ICAPA = (KPA * (MPA(OI, COI) - 10!) / 10!) - SUBCAPA
        ' Lower primary air (gas is now off -- less demand)
        flag2 = 1
END IF


' Add actions to previous values to get absolute control valves.


CAFR = CAFR + ICAFR
```

```
CASA = CASA + ICASA

IF CASA < .5 OR CASA > 29 THEN

        CAPA = CAPA + ICAPA

ELSE

        ICAPA = ICAPA - ICASA          ' Decoupling action

        CAPA = CAPA + ICAPA

END IF



IF CAFR < 0 THEN CAFR = 0

IF CASA < .5 THEN CASA = .5

IF CAPA < 9 THEN CAPA = 9


CALL inter      ' Interpolate to find 12-bit control words


' Print changes in oxy. and temp. and control actions


LOCATE 10, 38: PRINT USING "##.##"; coxy

LOCATE 10, 64: PRINT USING "###.#"; CTEMPS

LOCATE 14, 25: PRINT USING "##.##"; CAFR

LOCATE 14, 42: PRINT USING "###.##"; ICAFR

LOCATE 15, 25: PRINT USING "##.##"; CAPA

LOCATE 15, 42: PRINT USING "###.##"; ICAPA

LOCATE 16, 25: PRINT USING "##.##"; CASA

LOCATE 16, 42: PRINT USING "###.##"; ICASA
```

```
Time.now2 = TIMER
Elapse = (Time.now2 - time.start) / 60
LOCATE 3, 11: PRINT USING "###.##"; Elapse

delta = Time.now2 - Time.now1
LOCATE 3, 38: PRINT USING "###.##"; delta

' Write data to file

IF print.flag = 1 THEN
        WRITE #1, Elapse, delay, TEMPS, CTEMPS, settemp, temp(1, 4), scfm(1),
                  scfm(2), gasp(1), coxy, setoxy, CAPA, CASA, CAFR, KPA, KSA,
                  KFR
        Dcount = Dcount + 1
        LOCATE 22, 8: PRINT "Data Point:                                    "
        LOCATE 22, 20: PRINT Dcount
END IF

GOTO 100

press:
'--------------------------------------------------------------------
'******************** READ PRESSURE TRANSDUCER ********************
'--------------------------------------------------------------------
```

```
adr% = 768
MODE% = 0: flag% = 0
CALL DAS8(MODE%, VARPTR(adr%), flag%)
IF flag% <> 0 THEN LOCATE 1, 5: PRINT "ERROR 1.0: DAS-8 INSTALLATION"

MODE% = 1: Lt%(0) = 4: Lt%(1) = 4: flag% = 0
CALL DAS8(MODE%, VARPTR(Lt%(0)), flag%)
IF flag <> 0 THEN LOCATE 22, 8: PRINT "Error 6.0: Pressure Sub"

     flag% = 0: pres% = 0
     press.sum = 0
     FOR N = 1 TO 50
         MODE% = 4
         CALL DAS8(MODE%, VARPTR(pres%), flag%)
         IF flag% <> 0 THEN LOCATE 22, 8: PRINT "ERROR 6.1 Pressure Sub"
         press.sum = press.sum + pres% / 2047!
     NEXT N
     inches.air = press.sum / 50 * 10
     inches.air = inches.air - zoff
     IF inches.air < 0 THEN inches.air = 0
     flow = pressure * (orif(ornum, 1)) * ((inches.air / pressure) ^
            orif(ornum, 2))

RETURN
```

Thermocouple:

```
'-------------------------------------------------------------------------
'****************** READ THERMOCOUPLES AND GAS ANALYZERS *******************
'-------------------------------------------------------------------------


' READ THERMOCOUPLES


adr% = 768
MODE% = 0: flag% = 0
CALL DAS8(MODE%, VARPTR(adr%), flag%)
IF flag% <> 0 THEN LOCATE 1, 5: PRINT "ERROR 1.0: DAS-8 INSTALLATION"


NTIMES = 50
FOR P = 1 TO 1
        FOR c = 1 TO 4
                TSUM(P, c) = 0
        NEXT c
NEXT P


FOR JJ = 1 TO NTIMES
        FOR board% = 1 TO 1
                MODE% = 1: Lt%(0) = board%: Lt%(1) = board%
                CALL DAS8(MODE%, VARPTR(Lt%(0)), flag%)
                IF flag% <> 0 THEN LOCATE 22, 8: PRINT "Error 7.0: Thermocouple
```

```
                                                Sub"
            FOR chan% = 1 TO 4
                    MODE% = 14
                    CALL DAS8(MODE%, VARPTR(chan%), flag%)
                    IF flag% <> 0 THEN LOCATE 22, 8: PRINT "Error 7.1:
                                                        Thermocouple
                                                        Sub"
                    MODE% = 4
                    CALL DAS8(MODE%, VARPTR(DIGITAL%(board%, chan%)), flag%)
                    IF flag% <> 0 THEN LOCATE 22, 8: PRINT "Error 7.2:
                                                        Thermocouple
                                                        Sub"
                    voltage(board%, chan%) = DIGITAL%(board%, chan%) * 5! /
                                                (GAIN(board%) * 2047!)
                    CALL interpolation(tf, voltage(), board%, chan%)
                    T(board%, chan%) = tf
                    TSUM(board%, chan%) = TSUM(board%, chan%) +
                                                T(board%, chan%)
            NEXT chan%
        NEXT board%
NEXT JJ

FOR b = 1 TO 1
        FOR c = 1 TO 4
                temp(b, c) = TSUM(b, c) / NTIMES
```

```
        NEXT c
NEXT b

' SELECT 1 OF 3 COMBUSTION BED TEMPERATURES TO BASE CONTROL ACTIONS ON.
' USE 1 OF THE TEMPERATURES THAT IS ASSOCIATED WITH THE SMALLEST
' TEMPERATURE DIFFERENCE (ASSUMPTION: ONLY ONE PROBE WILL BE BAD AT A TIME).

'D12 = ABS(temp(1, 1) - temp(1, 2))
'D13 = ABS(temp(1, 1) - temp(1, 3))
'D23 = ABS(temp(1, 2) - temp(1, 3))

'DSMALL = D12
'IF D13 < DSMALL THEN DSMALL = D13
'IF D23 < DSMALL THEN DSMALL = D23

'IF DSMALL = D12 THEN TEMPS = temp(1, 2): PROBE = 1
'IF DSMALL = D13 THEN TEMPS = temp(1, 3): PROBE = 3
'IF DSMALL = D23 THEN TEMPS = temp(1, 2): PROBE = 2
TEMPS = temp(1, 2): PROBE = 2

' READ GAS ANALYZERS

adr% = 832
MODE% = 0: flag% = 0
CALL DAS8(MODE%, VARPTR(adr%), flag%)
```

```
IF flag% <> 0 THEN LOCATE 1, 5: PRINT "ERROR 1.0: DAS-8 INSTALLATION"

XXX = 20

FOR chan% = 1 TO 5
        MODE% = 1: Lt%(0) = chan%: Lt%(1) = chan%
        CALL DAS8(MODE%, VARPTR(Lt%(0)), flag%)
        MODE% = 4: SUM2 = 0
        FOR MM = 1 TO XXX
             CALL DAS8(MODE%, VARPTR(conc%), flag%)
             IF flag% <> 0 THEN LOCATE 22, 8: PRINT "Error 7.3: Thermocouple
                                                         Sub"
             SUM2 = SUM2 + conc%
        NEXT MM
        gasp(chan%) = Range(chan%) * SUM2 * 5 / (XXX * VOLT(chan%) * 2047!)
        GASUM(chan%) = GASUM(chan%) + gasp(chan%)
NEXT chan%
RETURN

keycode:
'-------------------------------------------------------------------
'********************** KEY DEFINITIONS ***************************
'-------------------------------------------------------------------

key1:
```

```
KEY(1) OFF: KEY(2) OFF: KEY(3) OFF: KEY(4) OFF: KEY(5) OFF: KEY(10) OFF
LOCATE 21, 8: PRINT "                                                  "
LOCATE 21, 8: PRINT "Old delay value (sec.): "; USING "##.##"; delay
LOCATE 22, 8: PRINT "                                                  "
LOCATE 22, 8: INPUT "Input new delay (sec.): ", delay
LOCATE 21, 8: PRINT "                                                  "
LOCATE 22, 8: PRINT "                                                  "
LOCATE 3, 38: PRINT "    "
GOTO keyend:


key2:
KEY(1) OFF: KEY(2) OFF: KEY(3) OFF: KEY(4) OFF: KEY(5) OFF: KEY(10) OFF
LOCATE 21, 8: PRINT "                                                  "
LOCATE 21, 8: PRINT "Old primary air scaling factor: "; USING "##.#"; KPA
LOCATE 22, 8: PRINT "                                                  "
LOCATE 22, 8: INPUT "Input new primary air scaling factor: ", KPA
LOCATE 21, 8: PRINT "                                                  "
LOCATE 21, 8: PRINT "Old secondary air scaling factor: "; USING "##.#"; KSA
LOCATE 22, 8: PRINT "                                                  "
LOCATE 22, 8: INPUT "Input new secondary air scaling factor: ", KSA
LOCATE 21, 8: PRINT "                                                  "
LOCATE 21, 8: PRINT "Old coal feed rate scaling factor: "; USING "##.#"; KFR
LOCATE 22, 8: PRINT "                                                  "
LOCATE 22, 8: INPUT "Input new coal feed rate scaling factor: ", KFR
LOCATE 21, 8: PRINT "                                                  "
```

```
LOCATE 22, 8: PRINT "                                              "
GOTO keyend:


key3:
KEY(1) OFF: KEY(2) OFF: KEY(3) OFF: KEY(4) OFF: KEY(5) OFF: KEY(10) OFF
LOCATE 21, 8: PRINT "                                              "
LOCATE 21, 8: PRINT "Old coal feed rate (lb/hr): "; USING "##.#"; frate
LOCATE 22, 8: PRINT "                                              "
LOCATE 22, 8: INPUT "Input new coal feed rate (lb/hr): ", frate
LOCATE 21, 8: PRINT "                                              "
LOCATE 22, 8: PRINT "                                              "
IF frate < 5 OR frate > 20 THEN GOTO key3:
GOTO keyend:


key4:
KEY(1) OFF: KEY(2) OFF: KEY(3) OFF: KEY(4) OFF: KEY(5) OFF: KEY(10) OFF
LOCATE 21, 8: PRINT "                                              "
LOCATE 21, 8: PRINT "Enter:  1) to stop data taking"
LOCATE 22, 8: PRINT "                                              "
LOCATE 22, 8: INPUT "        2) to restart data taking ", sel
LOCATE 21, 8: PRINT "                                              "
LOCATE 22, 8: PRINT "                                              "
IF sel = 1 THEN
        print.flag = 0
        GOTO keyend:
```

```
END IF
IF sel = 2 THEN
        print.flag = 1
        GOTO keyend:
END IF
GOTO key4:

key5:
KEY(1) OFF: KEY(2) OFF: KEY(3) OFF: KEY(4) OFF: KEY(5) OFF: KEY(10) OFF
LOCATE 19, 8: PRINT "                                                  "
LOCATE 20, 8: PRINT "                                                  "
LOCATE 21, 8: PRINT "                                                  "
LOCATE 22, 8: PRINT "                                                  "
IF flag4 = 1 THEN
        flag4 = 0
        CSUM = CSUM - 2 ^ 5     ' Turn off red warning light
        OUT &H305, CSUM
END IF
GOTO keyend:

key10:
KEY(1) OFF: KEY(2) OFF: KEY(3) OFF: KEY(4) OFF: KEY(5) OFF: KEY(10) OFF
IF print.flag = 1 THEN
        LOCATE 22, 8: PRINT "                                                  "
        LOCATE 22, 8: PRINT "Data collection must be halted before exiting."
```

```
        GOTO keyend:
END IF
CLOSE #1
CLS : PRINT "PROGRAM TERMINATED"
OUT &H305, 0
FOR N = 0 TO 3
OUT &H350 + 2 * N, 0
OUT &H350 + 1 + 2 * N, 0
NEXT N
STOP: END
GOTO keyend:


keyend:
KEY(1) ON: KEY(2) ON: KEY(3) ON: KEY(4) ON: KEY(5) ON: KEY(10) ON
RETURN
END




SUB display


CLS


'----------------------------------------------------------------------
```

```
'****************** SCREEN HEADINGS OVERLAY SUBROUTINE ****************
'----------------------------------------------------------------------

LOCATE 1, 24: PRINT " FUZZY LOGIC CONTROLLER "

LOCATE 3, 1: PRINT "Elapse:"
LOCATE 3, 18: PRINT "min."
LOCATE 3, 29: PRINT "Delta:"
LOCATE 3, 45: PRINT "sec."
LOCATE 3, 58: PRINT "Delay:"
LOCATE 3, 74: PRINT "sec."

LOCATE 5, 1: PRINT "Temperature (F):   Probe # 1      Probe # 2      Probe # 3
                    Probe In Use"
LOCATE 6, 1: PRINT "----------------------------------------------------------
                    ------------"
LOCATE 7, 1: PRINT "Central Bed:"
LOCATE 8, 1: PRINT "Annular Bed:"
LOCATE 8, 40: PRINT "-"
LOCATE 8, 55: PRINT "-"
LOCATE 8, 71: PRINT "1"

LOCATE 10, 1: PRINT "Oxygen (%):        Oxy. Change (%):        Temp. Change
                     (F):"
```

```
LOCATE 12, 1: PRINT "Control actions:       Absolute       Last Increment
                    Measured Values:"
LOCATE 13, 1: PRINT "----------------------------------------------------
                    ----------------"
LOCATE 14, 1: PRINT "Coal Feed Rate (lb/hr):"
LOCATE 14, 69: PRINT "-"
LOCATE 15, 1: PRINT "Pri. Air Flow   (scfm):"
LOCATE 16, 1: PRINT "Sec. Air Flow   (scfm):"

LOCATE 18, 57: PRINT "F[1] ... Delay"
LOCATE 19, 57: PRINT "F[2] ... Scaling Factors"
LOCATE 20, 57: PRINT "F[3] ... Feed Rate"
LOCATE 21, 57: PRINT "F[4] ... Data Collection"
LOCATE 22, 57: PRINT "F[5] ... Clear MSGS"
LOCATE 23, 57: PRINT "F[10] .. Exit"

LOCATE 18, 1: PRINT "------------------------------------------------------|"
LOCATE 19, 1: PRINT "|MSGS:                                                |"
LOCATE 20, 1: PRINT "|                                                     |"
LOCATE 21, 1: PRINT "|                                                     |"
LOCATE 22, 1: PRINT "|                                                     |"
LOCATE 23, 1: PRINT "|-----------------------------------------------------|"

END SUB
```

```
SUB inter

' Interpolate to convert control actions into 12-bit format.

IF CAFR = 0 THEN
        D(0) = 0
        GOTO oca:
END IF
FOR X = 1 TO 20
IF CAFR >= FEED(X) AND CAFR < FEED(X + 1) GOTO INTP:
IF CAFR > FEED(X) AND FEED(X + 1) = 0 THEN
        LOCATE 21, 8: PRINT "ERROR: SATURATION IN FEED RATE CONVERSION."
        D(0) = FD(X)
        CAFR = CAFR - ICAFR
        GOTO oca:
END IF
NEXT X

INTP:
D(0) = FD(X + 1) - ((FEED(X + 1) - CAFR) / (FEED(X + 1) - FEED(X))) *
       (FD(X + 1) - FD(X))
```

```
oca:
IF CAPA = 0 THEN
        D(2) = 0
        GOTO oca1:
END IF
FOR X = 1 TO 81
IF CAPA >= PFLOW(X) AND CAPA < PFLOW(X + 1) GOTO INTP1:
IF CAPA > PFLOW(X) AND PFLOW(X + 1) = 0 THEN
        LOCATE 21, 8: PRINT "ERROR: SATURATION IN PRIMARY AIR CONVERSION."
        D(2) = PD(X)
        CAPA = CAPA - ICAPA
        GOTO oca1:
END IF
NEXT X

INTP1:
D(2) = PD(X + 1) - ((PFLOW(X + 1) - CAPA) / (PFLOW(X + 1) - PFLOW(X))) *
       (PD(X + 1) - PD(X))

oca1:
IF CASA = 0 THEN
        D(3) = 0
        GOTO oca2:
END IF
FOR X = 1 TO 81
```

```
IF CASA >= SFLOW(X) AND CASA < SFLOW(X + 1) GOTO INTP2:
IF CASA > SFLOW(X) AND SFLOW(X + 1) = 0 THEN
        LOCATE 21, 8: PRINT "ERROR: SATURATION IN SECONDARY AIR CONVERSION."
        D(3) = SD(X)
        CASA = CASA - ICASA
        GOTO oca2:
END IF
NEXT X

INTP2:
D(3) = SD(X + 1) - ((SFLOW(X + 1) - CASA) / (SFLOW(X + 1) - SFLOW(X))) *
       (SD(X + 1) - SD(X))

oca2:
D(0) = CINT(D(0))
D(2) = CINT(D(2))
D(3) = CINT(D(3))

' Output control actions

FOR N = 0 TO 3
XH% = INT(D(N) / 256)
XL% = D(N) - XH% * 256
OUT &H350 + 2 * N, XL%
OUT &H350 + 1 + 2 * N, XH%
```

```
NEXT N

END SUB

SUB interpolation (tf, voltage(), board%, chan%)

'-------- Interpolation routine to find K thermocouple temperature -----
'Entry variables:-
'    CJC = cold junction compensator temperature in deg. C.
'      VOLT = thermocouple voltage in volts
'Exit variables:-
'     TC = temperature in degrees Centigrade
'     TF = temperature in degrees Fahrenheit
'Execution time on std. IBM P.C. = 46 milliseconds
'Perform CJC compensation for K type

vk = 1000 * voltage(board%, chan%) + 1! + (cjc - 25) * .0405   'VK in mV

'Find look up element
EK = INT((vk - svk) / sik)
'Out of bounds, round to lower limit
IF EK < 0 THEN Tc = Tk(0): GOTO 2360
```

```
'Out of bounds, round to upper limit
IF EK > nk - 2 THEN Tc = Tk(nk - 1): GOTO 2360
'Do interpolation
Tc = Tk(EK) + (Tk(EK + 1) - Tk(EK)) * (vk - EK * sik - svk) / sik   'Centigrade

2360 tf = Tc * 9 / 5 + 32                                          'Fahrenheit
END SUB




SUB title

'--------------------------------------------------------------------
'*************************** TITLE **********************************
'--------------------------------------------------------------------

CLS
 LOCATE 5, 20: PRINT "|------------------------------------|";
 LOCATE 6, 20: PRINT "|                                    |";
 LOCATE 7, 20: PRINT "|  *****  *   *  *****  *****  *   * |";
 LOCATE 8, 20: PRINT "|  *      *   *     *      *    * *  |";
 LOCATE 9, 20: PRINT "|  ****   *   *    *      *      *   |";
LOCATE 10, 20: PRINT "|  *      *   *   *      *       *   |";
LOCATE 11, 20: PRINT "|  *       ***   *****  *****    *   |";
```

```
LOCATE 12, 20: PRINT "|                                     |";
LOCATE 13, 20: PRINT "|       LOGIC CONTROL SOFTWARE        |";
LOCATE 14, 20: PRINT "|    FOR FLUIDIZED BED COMBUSTORS     |";
LOCATE 15, 20: PRINT "|                                     |";
LOCATE 16, 20: PRINT "|                                     |";
LOCATE 17, 20: PRINT "|         (c) 1989 Fuzzyware          |";
LOCATE 18, 20: PRINT "|                                     |";
LOCATE 19, 20: PRINT "|-------------------------------------|";
LOCATE 22, 26: PRINT "Press any key to continue..."

DO:LOOP UNTIL INKEY$ <> ""

END SUB
```

8.2 Sample FORTRAN Code

FORTRAN code for determining adjustments to secondary air flow using fuzzy reasoning is shown below; code for all other manipulated variables is similar. In this code, fuzzy sets are generically represented as arrays with elements ranging from 0 to 20. Assignment of values to elements (or scaling) occurs in the QuickBASIC control code.

```
*******************************************************************
*                                                                 *
*       FUZZY LOGIC LOOK-UP TABLE FORMULATION FOR                 *
*       SECONDARY AIR                                             *
*                                                                 *
*       09/10/88                                                  *
*                                                                 *
*******************************************************************


        REAL LOW(0:20),OK(0:20),HIGH(0:20),POS(0:20),ZERO(0:20),
     &  NEG(0:20),NPOS(0:20),NNEG(0:20),MORE(0:20),SAME(0:20),
     &  LESS(0:20),UTEMP(7,0:20),UCHANG(7,0:20),USEC(7,0:20),
     &  UUSEC(0:20),UR(7,0:20,0:20,0:20),URT(0:20,0:20,0:20),
     &  CASEC(0:20,0:20)

C       DEFINITION OF LOW FOR TEMP. OF COMBUSTION BED
```

```
      LOW(0)=1.0
      LOW(1)=0.9
      LOW(2)=0.8
      LOW(3)=0.7
      LOW(4)=0.6
      LOW(5)=0.5
      LOW(6)=0.4
      LOW(7)=0.3
      LOW(8)=0.2
      LOW(9)=0.1
      LOW(10)=0.0
      LOW(11)=0.0
      LOW(12)=0.0
      LOW(13)=0.0
      LOW(14)=0.0
      LOW(15)=0.0
      LOW(16)=0.0
      LOW(17)=0.0
      LOW(18)=0.0
      LOW(19)=0.0
      LOW(20)=0.0

C     DEFINITION OF OK FOR TEMP. OF COMBUSTION BED
      OK(0)=0.0
      OK(1)=0.1
```

```
      OK(2)=0.2
      OK(3)=0.3
      OK(4)=0.4
      OK(5)=0.5
      OK(6)=0.6
      OK(7)=0.7
      OK(8)=0.8
      OK(9)=0.9
      OK(10)=1.0
      OK(11)=0.9
      OK(12)=0.8
      OK(13)=0.7
      OK(14)=0.6
      OK(15)=0.5
      OK(16)=0.4
      OK(17)=0.3
      OK(18)=0.2
      OK(19)=0.1
      OK(20)=0.0

C     DEFINITION OF HIGH FOR TEMP. OF COMBUSTION BED
      HIGH(0)=0.0
      HIGH(1)=0.0
      HIGH(2)=0.0
      HIGH(3)=0.0
```

```
      HIGH(4)=0.0
      HIGH(5)=0.0
      HIGH(6)=0.0
      HIGH(7)=0.0
      HIGH(8)=0.0
      HIGH(9)=0.0
      HIGH(10)=0.0
      HIGH(11)=0.1
      HIGH(12)=0.2
      HIGH(13)=0.3
      HIGH(14)=0.4
      HIGH(15)=0.5
      HIGH(16)=0.6
      HIGH(17)=0.7
      HIGH(18)=0.8
      HIGH(19)=0.9
      HIGH(20)=1.0

C     DEFINITION OF NEGATIVE FOR CHANGE-IN-TEMP.
      DO 30 J=0,20
30    NEG(J)=LOW(J)

C     DEFINITION OF NOT NEGATIVE FOR CHANGE-IN-TEMP.
      DO 50 J=0,20
50    NNEG(J)=1.0-NEG(J)
```

```
C       DEFINITION OF ZERO FOR CHANGE-IN-TEMP.
        DO 20 J=0,20
20      ZERO(J)=OK(J)

C       DEFINITION OF POSITIVE FOR CHANGE-IN-TEMP.
        DO 10 J=0,20
10      POS(J)=HIGH(J)

C       DEFINITION OF NOT POSITIVE FOR CHANGE-IN-TEMP.
        DO 40 J=0,20
40      NPOS(J)=1.0-POS(J)

C       DEFINITION OF LESS FOR SECONDARY AIR
        DO 60 J=0,20
60      LESS(J)=LOW(J)

C       DEFINITION OF SAME FOR SECONDARY AIR
        DO 70 J=0,20
70      SAME(J)=OK(J)

C       DEFINITION OF MORE FOR SECONDARY AIR
        DO 80 J=0,20
80      MORE(J)=HIGH(J)
```

```
C       ASSIGNMENT OF THE CONTROL RULES

C       RULE #1
        CALL EQUATE(UTEMP,LOW,1)
        CALL EQUATE(UCHANG,POS,1)
        CALL EQUATE(USEC,SAME,1)

C       RULE #2
        CALL EQUATE(UTEMP,LOW,2)
        CALL EQUATE(UCHANG,NPOS,2)
        CALL EQUATE(USEC,LESS,2)

C       RULE #3
        CALL EQUATE(UTEMP,OK,3)
        CALL EQUATE(UCHANG,POS,3)
        CALL EQUATE(USEC,MORE,3)

C       RULE #4
        CALL EQUATE(UTEMP,OK,4)
        CALL EQUATE(UCHANG,ZERO,4)
        CALL EQUATE(USEC,SAME,4)

C       RULE #5
```

```
      CALL EQUATE(UTEMP,OK,5)
      CALL EQUATE(UCHANG,NEG,5)
      CALL EQUATE(USEC,LESS,5)

C     RULE #6
      CALL EQUATE(UTEMP,HIGH,6)
      CALL EQUATE(UCHANG,NNEG,6)
      CALL EQUATE(USEC,MORE,6)

C     RULE #7
      CALL EQUATE(UTEMP,HIGH,7)
      CALL EQUATE(UCHANG,NEG,7)
      CALL EQUATE(USEC,SAME,7)


      TYPE*,' '
      TYPE*,'COMPUTATIONS TAKING PLACE -- PLEASE WAIT.'

C     COMPUTATION OF G.O.M. OF INDIVIDUAL CONTROL RULES
      DO 160 R=1,7
      DO 160 I=0,20
      DO 160 J=0,20
      DO 160 K=0,20
      UR(R,I,J,K)=AMIN1(UTEMP(R,I),UCHANG(R,J),USEC(R,K))
```

```
160     CONTINUE

C       COMPUTATION OF G.O.M. OF CONTROL LAW (UNION OF INDIVIDUAL CONTROL RULES)
        DO 310 I=0,20
        DO 310 J=0,20
        DO 310 K=0,20
        COMP=0.0
        DO 310 R=1,7
        URT(I,J,K)=AMAX1(UR(R,I,J,K),COMP)
        COMP=URT(I,J,K)
310     CONTINUE

C       COMPUTATION OF CONTROLLER OUTPUT (FUZZY FORM)
        DO 200 I=0,20
        DO 200 J=0,20
        DO 210 K=0,20
        UUSEC(K)=URT(I,J,K)             ! INPUT TO FUZZY MATRIX
210     CONTINUE                        ! IS FUZZY SINGLETON

C       CENTER-OF-AREA DEFUZZIFICATION ROUTINE
        SUMN=0.0
        SUMD=0.0
        DO 250 K=0,20
        SUMN=SUMN+(K*UUSEC(K))
        SUMD=SUMD+(UUSEC(K))
```

```
250     CONTINUE
        CASEC(I,J)=(SUMN/SUMD)
200     CONTINUE

        TYPE*,' '
        TYPE*,'LOOK-UP TABLE FORMULATION FOR SECONDARY AIR IS NOW FINISHED.'
        TYPE*,'THIS TABLE IS BEING OUTPUTED TO THE FILE SEC.FLC.'

        OPEN(UNIT=7,IOSTAT=IOSTAT,ERR=1000,FILE='SEC.FLC',STATUS='NEW',
     &  ACCESS='SEQUENTIAL',FORM='FORMATTED',BLANK='ZERO')
        WRITE(7,510)CASEC
510     FORMAT(F16.4)
        CLOSE(UNIT=7,IOSTAT=IOSTAT,ERR=1025,STATUS='KEEP')

        GO TO 1050

1000    TYPE*,' '
        TYPE*,'AN ERROR HAS OCCURRED DURING A FILE OPENING OPERATION.'
        TYPE*,'IOSTAT=',IOSTAT
        GO TO 1050
1025    TYPE*,' '
        TYPE*,'AN ERROR HAS OCCURRED DURING A FILE CLOSING OPERATION.'
        TYPE*,'IOSTAT=',IOSTAT

1050    STOP
```

```
      END

      SUBROUTINE EQUATE(A,B,R)
      INTEGER R
      REAL A(7,0:20),B(0:20)

      DO 500 J=0,20
500   A(R,J)=B(J)

      RETURN
      END
```

9. APPENDIX B. COMBUSTOR SIMULATIONS

Nomenclature

AB	=	subscript referring to annular bed
AG	=	subscript referring to annular bed gas
AP	=	subscript referring to annular bed particles
A_{CB}	=	heat transfer area of combustion bed
a	=	surface area of sand per volume of gas in annular bed
CB	=	subscript referring to combustion bed
$C_b(z,t)$	=	bubble phase oxygen concentration
$\bar{C}_b(t)$	=	average of $C_b(t)$ over z-direction (vertical)
$C_e(t)$	=	emulsion phase oxygen concentration
C_o	=	inlet oxygen concentration to emulsion phase evaluated at T_{CB}
C_p	=	specific heat
D	=	diameter of fuel particle
D_{CB}	=	diameter of combustion bed
D_G	=	diffusivity of oxygen
$E(t)$	=	heat generation rate

$F(D,t)$ = feed rate for fuel particle size between D and ΔD

$F_T(t)$ = total feed rate

$f(t)$ = combustion rate coefficient

H_c = net heat of combustion

h_{CB} = heat transfer coefficient between combustion bed and inner combustor wall

h_{GP} = heat transfer coefficient (HTC) between gas and particles

h_{GW} = HTC between gas and wall surfaces

h_{PW} = HTC between particles and wall surfaces

h_W = HTC between water and water jacket wall

IW = subscript referring to inner combustor wall

$K(t)$ = fuel consumption rate

k_I = exchange coefficient between bubble and emulsion phase

L_F = fluidized combustion bed height

$M(t)$ = fuel mass present in combustor

o = subscript referring to inlet air conditions

P = subscript referring to particles in combustion bed

$\dot{q}$ = heat generation rate per unit fluidized combustion bed volume

q'' = heat flux from water jacket wall to water

q_b'' = subcooled boiling heat flux

r = oxygen consumption rate

Sh = Sherwood number

T	=	temperature
$\bar{T}$	=	average temperature
t	=	time
U_{AG}	=	velocity of gas passing upward through annular bed
U_B	=	bubble rise velocity
U_{mf}	=	minimum fluidization velocity
U_o	=	inlet air velocity to combustion bed
U_W	=	water velocity in water jacket
W	=	subscript referring to water
WW	=	subscript referring to water jacket wall
z	=	vertical dimension
ΔD	=	change in fuel particle diameter
Δx	=	annular thickness
ε	=	voidage
ε_{mf}	=	voidage at minimum fluidization
ϵ	=	emissivity
$\lambda(D,t)$	=	fuel mass present in combustor in the interval D to ΔD
ρ	=	density
ρ_c	=	density of carbon in coal
σ	=	Stefan-Boltzmann constant

A brief description of the combustor model is first given; simulations are then discussed.

9.1 Combustor Model

The combustor is simulated by numerically integrating differential equations describing a combustion model and a heat transfer model. The combustion model simulates heat release from the fuel and consumption of oxygen during the process, and the heat transfer model describes time-temperature relations of the combustor. Both models contain partial differential equations which are transformed by the method of lines [25] into systems of ordinary differential equations.

The combustion model is similar to that suggested by Van der Post *et al.* [26]. Several assumptions are used in this formulation: no attrition or elutriation of fuel particles occur; fuel particles are spherical; and $C + O_2 \rightarrow CO_2$ is the only chemical reaction. The partial differential equation describing mass distribution of fuel in the combustor is

$$\frac{\partial}{\partial t}[\lambda(D,t)] - \frac{f(t)}{D}\frac{\partial}{\partial D}[\lambda(D,t)] = F(D,t) - \frac{4f(t)}{D^2}\lambda(D,t).$$

The total mass present in the combustor is then

$$M(t) = \sum_D \lambda(D,t).$$

Also, the total feed rate is

$$F_T(t) = \sum_D F(D,t).$$

The combustion rate coefficient $f(t)$ appearing in the mass distribution equation is a function of emulsion phase oxygen concentration $C_e(t)$:

$$f(t) = \frac{48ShD_G}{\rho_c} C_e(t).$$

This oxygen concentration is found by integrating the equations:

$$\begin{aligned}
\frac{\partial}{\partial t}[C_b(z,t)] + U_B \frac{\partial}{\partial z}[C_b(z,t)] &= k_I[C_e(t) - C_b(z,t)] \\
\frac{d}{dt}[C_e(t)] &= \frac{U_{mf}}{L_f \varepsilon_{mf}}[C_o(t) - C_e(t)] + \frac{k_I}{\varepsilon_{mf}}[\bar{C}_b(t) - C_e(t)] \\
&\quad - \frac{r}{A_{CB} L_{mf} \varepsilon_{mf}}.
\end{aligned}$$

Total fuel consumption rate is obtained by summing combustion rates of individual fuel diameter intervals:

$$K(t) = \sum_D \frac{3f(t)}{D^2} \lambda(D,t).$$

The heat generated per unit time is

$$E(t) = H_c \times K(t).$$

Heating value and elemental analysis of the fuel were taken to be those of Illinois No. 5 Rapattee coal (see Table 9.1), a typical fuel used during experimental tests.

The heat transfer model is developed from the following assumptions. The combustion bed is at uniform temperature. Heat of combustion is released uniformly to the combustion bed. Separate temperatures are calculated for gas and particles in the annular fluidized bed. The particles are assumed to be well mixed whereas the gas is assumed to be in plug flow. The mathematical formulation consists of partial differential equations describing heat transfer in the beds and the water jacket due to convection, radiation, and, in the case of water, subcooled boiling. The complete set of equations describing heat transfer are given below.

Table 9.1: Properties of Illinois No. 5 Coal

Proximate Analysis (as received):	
Moisture	10.8%
Ash	10.1%
Volatile Matter	36.0%
Fixed Carbon	42.5%
BTU/lb	11,395
Ultimate Analysis (dry basis):	
Carbon	77.88%
Hydrogen	3.76%
Nitrogen	1.49%
Oxygen	2.57%
Sulfur	3.00%
Ash	11.30%

- *Combustion Bed Temperature:*

$$\begin{aligned}\frac{d}{dt}[T_{CB}] &= \frac{1}{\rho_P c_{p_P}(1-\varepsilon_{CB})}\Bigg[\dot{q} + \frac{\rho_o c_{p_o} U_o}{L_F}(T_o - T_{CB}) \\ &\quad + \frac{4}{D_{CB}} h_{CB}(T_{IW} - T_{CB}) \\ &\quad + \frac{4}{D_{CB}}\sigma\left(\frac{1}{\epsilon_{CB}} + \frac{1}{\epsilon_{IW}} - 1\right)^{-1}(T_{IW}^4 - T_{CB}^4)\Bigg]\end{aligned}$$

- *Inner Combustor Wall Temperature:*

$$\begin{aligned}\frac{d}{dt}[T_{IW}] &= \frac{1}{(\rho c_p \Delta x)_{IW}}\Bigg[h_{CB}(T_{CB} - T_{IW}) + h_{GW}(\bar{T}_{AG} - T_{IW}) \\ &\quad + h_{PW}(T_{AP} - T_{IW}) + \sigma\left(\frac{1}{\epsilon_{CB}} + \frac{1}{\epsilon_{IW}} - 1\right)^{-1}(T_{CB}^4 - T_{IW}^4) \\ &\quad + \sigma\left(\frac{1}{\epsilon_{AP}} + \frac{1}{\epsilon_{IW}} - 1\right)^{-1}(T_{AP}^4 - T_{IW}^4)\Bigg]\end{aligned}$$

- *Annular Bed Gas Temperature:*

$$\begin{aligned}\frac{\partial}{\partial t}[T_{AG}] + \frac{U_{AG}}{\varepsilon_{AB}}\frac{\partial}{\partial z}[T_{AG}] &= \frac{1}{(\rho c_p \Delta x)_{AG}\ \varepsilon_{AB}}\Big[h_{GW}(T_{IW} - T_{AG}) \\ &\quad +\Delta x_{AG}\ \varepsilon_{AB}\ a\ h_{GP}(T_{AP} - T_{AG}) \\ &\quad +h_{GW}(T_{WW} - T_{AG})\Big]\end{aligned}$$

- *Annular Bed Particle Temperature:*

$$\begin{aligned}\frac{d}{dt}[T_{AP}] &= \frac{1}{(\rho c_p \Delta x)_{AP}(1-\varepsilon_{AB})}\Big[h_{PW}(T_{IW} - T_{AP}) + h_{GP}(\bar{T}_{AG} - T_{AP}) \\ &\quad +h_{PW}(T_{WW} - T_{AP}) + \sigma\left(\frac{1}{\epsilon_{AP}} + \frac{1}{\epsilon_{IW}} - 1\right)^{-1}(T_{IW}^4 - T_{AP}^4) \\ &\quad +\sigma\left(\frac{1}{\epsilon_{AP}} + \frac{1}{\epsilon_{WW}} - 1\right)^{-1}(T_{WW}^4 - T_{AP}^4)\Big]\end{aligned}$$

- *Water Jacket Wall Temperature:*

$$\begin{aligned}\frac{d}{dt}[T_{WW}] &= \frac{1}{(\rho c_p \Delta x)_{WW}}\Big[h_{PW}(T_{AP} - T_{WW}) + h_{GW}(\bar{T}_{AG} - T_{WW}) \\ &\quad +q'' + \sigma\left(\frac{1}{\epsilon_{WW}} + \frac{1}{\epsilon_{AP}} - 1\right)^{-1}(T_{AP}^4 - T_{WW}^4)\Big] \\ \text{where } q'' &= \begin{cases} h_W(T_{WW} - T_W), & T_{WW} \leq 100°C \\ q_b'', & T_{WW} > 100°C \end{cases}\end{aligned}$$

- *Water Temperature:*

$$\frac{\partial}{\partial t}[T_W] + U_W\frac{\partial}{\partial z}[T_W] = \frac{1}{(\rho c_p \Delta x)_W}\ q''$$

Convection coefficients associated with heat transfer between particles and gas and to surfaces are calculated from the two-phase theory of fluidization as described by Xavier and Davidson [27].

9.2 Simulations

Simulations of above model with the preliminary fuzzy controller are described in [21]. Results are reiterated in Figures 9.1, 9.2, and 9.3. Figure 9.1 shows combustion bed temperature and secondary air flow. Figure 9.2 depicts coal feed rate and coal consumption rate. Figure 9.3 illustrates flue gas oxygen and primary air flow rate. These figures are to be compared with actual system responses in Figures 5.1, 5.3, and 5.2, respectively. The model qualitatively compares favorably to the actual system, but quantitative comparisons are not as good.

For good quantitative comparisons, the model's steady-state values for each variable should be near those of the actual combustor, regardless of the controller used on the model and combustor. Being approximately 12% lower, the model's primary air flow compares well to the combustor's primary air flow. However, secondary air flow for the model is much less than that for the combustor. This is attributed to the model predicting fluidization in the annular bed at a lower velocity than that actually observed. With increased heat transfer associated with fluidization, smaller secondary air flow rates are predicted to maintain combustion bed temperature at 1600°F. The original output membership functions for secondary air had to be scaled down to reduce the effective secondary air flow gains such that the simulated system would be stable. Hence, the model was worthless for tuning the temperature control loop but served well for tuning the oxygen control loop. The majority of temperature control tuning occurred during actual test runs.

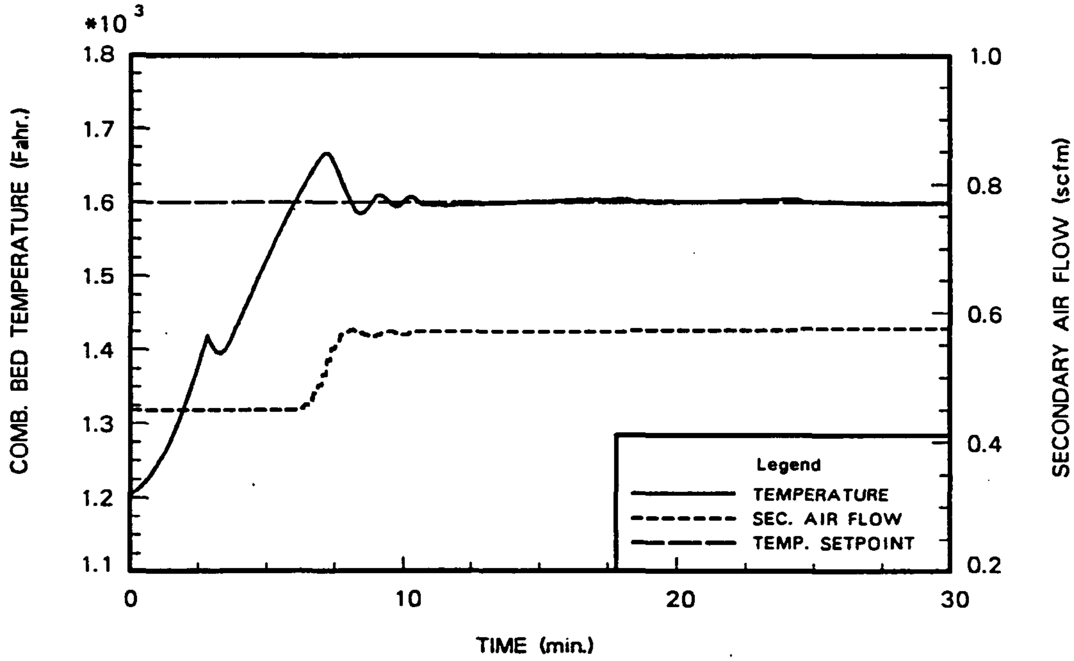

Figure 9.1: Simulated Combustion Bed Temperature and Secondary Air Flow Rate

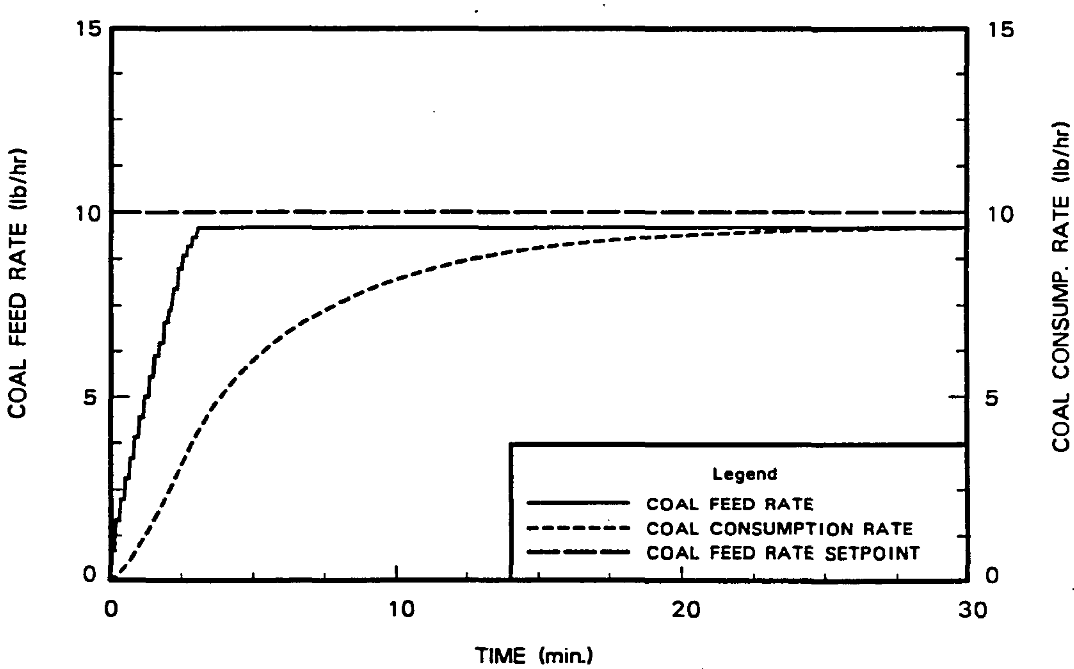

Figure 9.2: Simulated Coal Feed Rate and Coal Consumption Rate

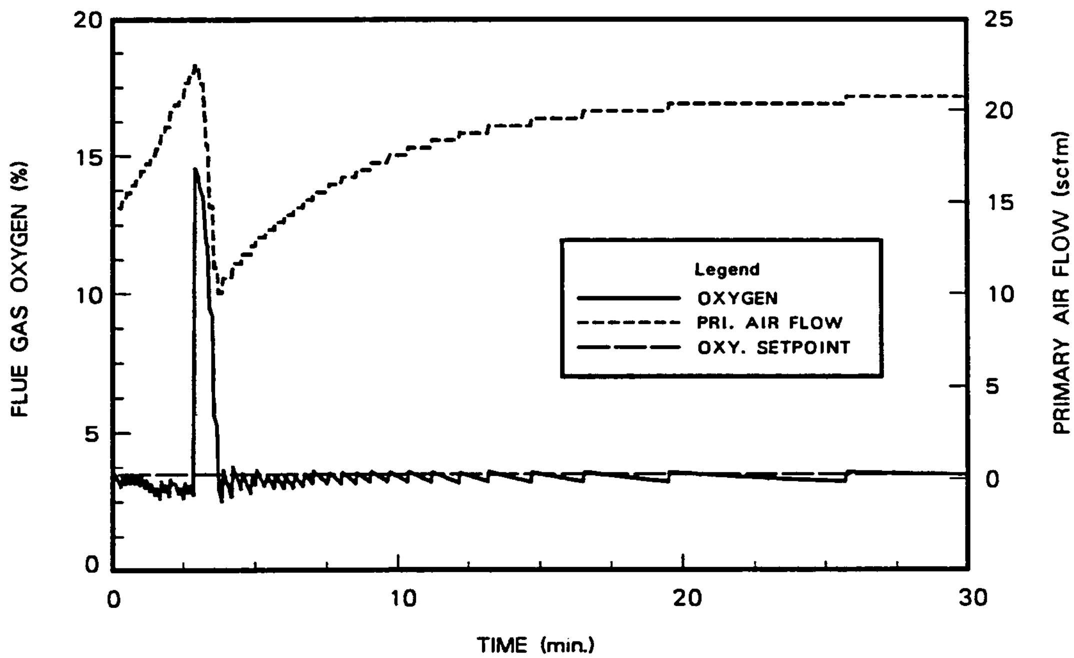

Figure 9.3: Simulated Flue Gas Oxygen and Primary Air Flow Rate

10. APPENDIX C. REDUCED RULE SETS

Table 10.1: Truth Table of Temperature Control Rules

T_{BED}	ΔT_{BED}	$\Delta Sec.\ Air\ Flow$
LOW	POSITIVE	NONE
LOW	NOT POSITIVE	LESS
OK	POSITIVE	MORE
OK	NEAR ZERO	NONE
OK	NEGATIVE	LESS
HIGH	NOT NEGATIVE	MORE
HIGH	NEGATIVE	NONE

Table 10.2: Truth Table of Flue Gas Oxygen Control Rules

$\%O_2$	$\Delta\%O_2$	$\Delta Pri.\ Air\ Flow$
LOW	POSITIVE	NONE
LOW	NOT POSITIVE	MORE
OK	POSITIVE	LESS
OK	NEAR ZERO	NONE
OK	NEGATIVE	MORE
HIGH	NOT NEGATIVE	LESS
HIGH	NEGATIVE	NONE

Table 10.3: Truth Table of Coal Feed Rate Control Rules

$\%O_2$[a]	$Error_{feed\ rate}$	$\Delta Coal\ Feed\ Rate$
LOW	POSITIVE	NONE
X	NEAR ZERO	NONE
X	NEGATIVE	LESS
NOT LOW	POSITIVE	MORE

[a]X denotes Don't Care conditions.

11. APPENDIX D. MEMBERSHIP FUNCTIONS

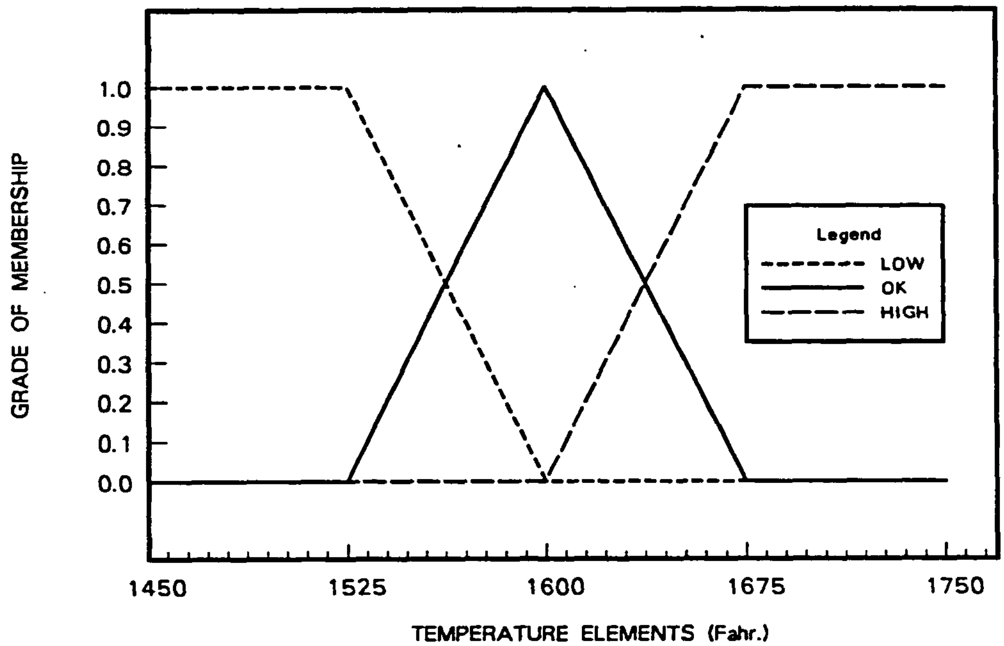

Figure 11.1: Membership Functions for "Low", "Ok", and "High" Temperature

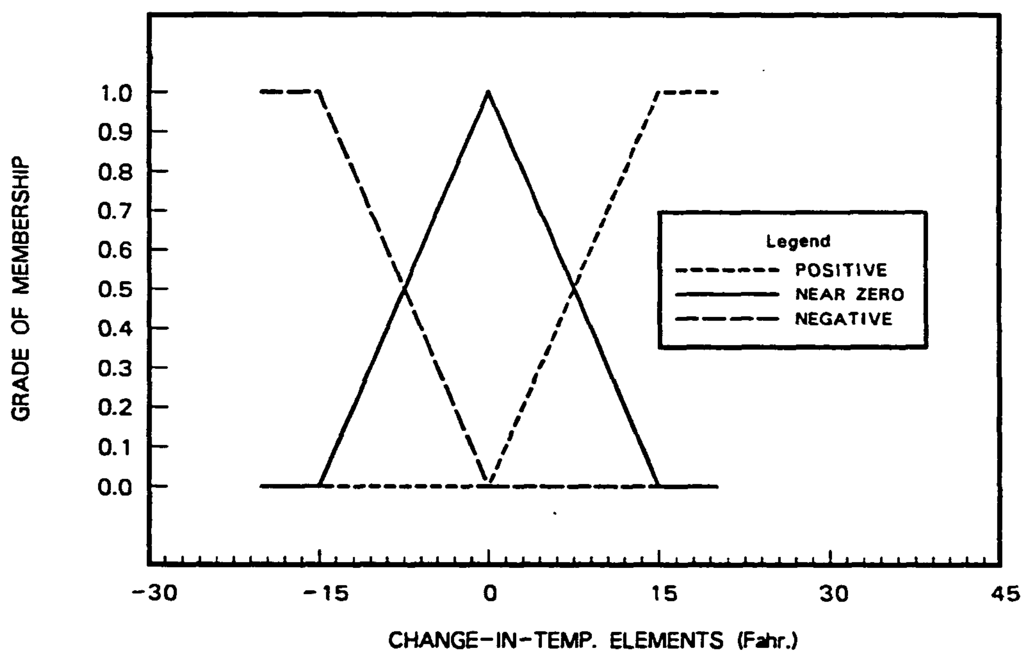

Figure 11.2: Membership Functions for "Positive", "Near Zero", and "Negative" Δ Temperature

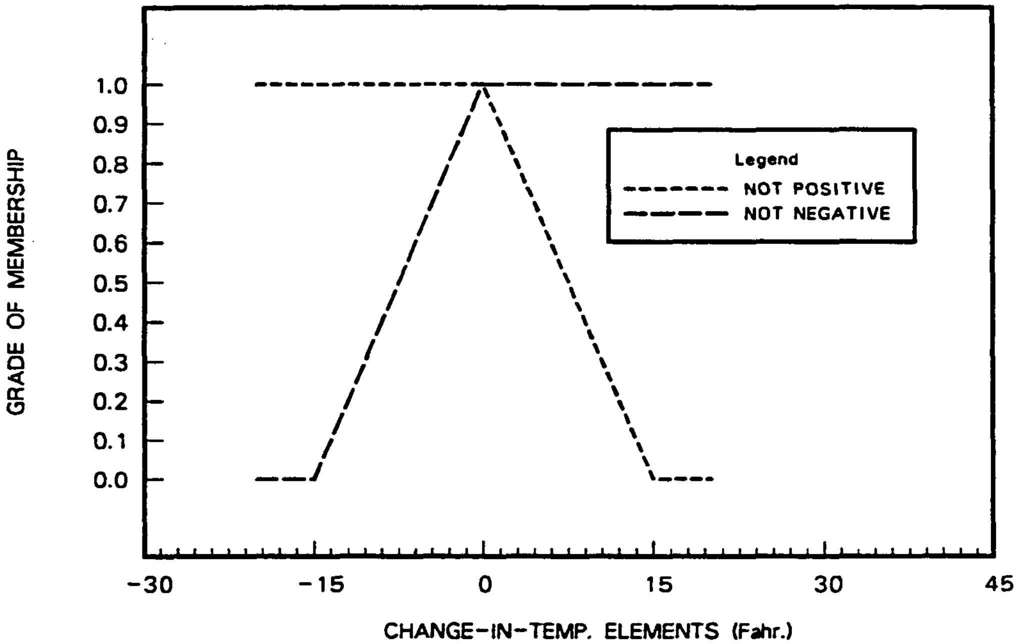

Figure 11.3: Membership Functions for "Not Positive" and "Not Negative" Δ Temperature

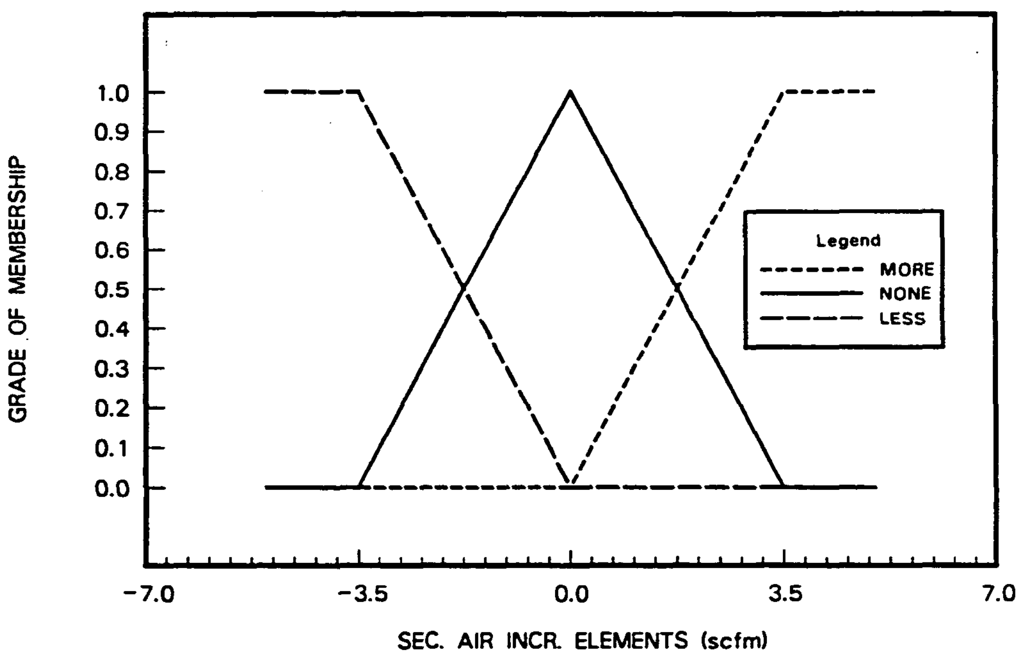

Figure 11.4: Membership Functions for "More", "None", and "Less" Secondary Air Flow Increment

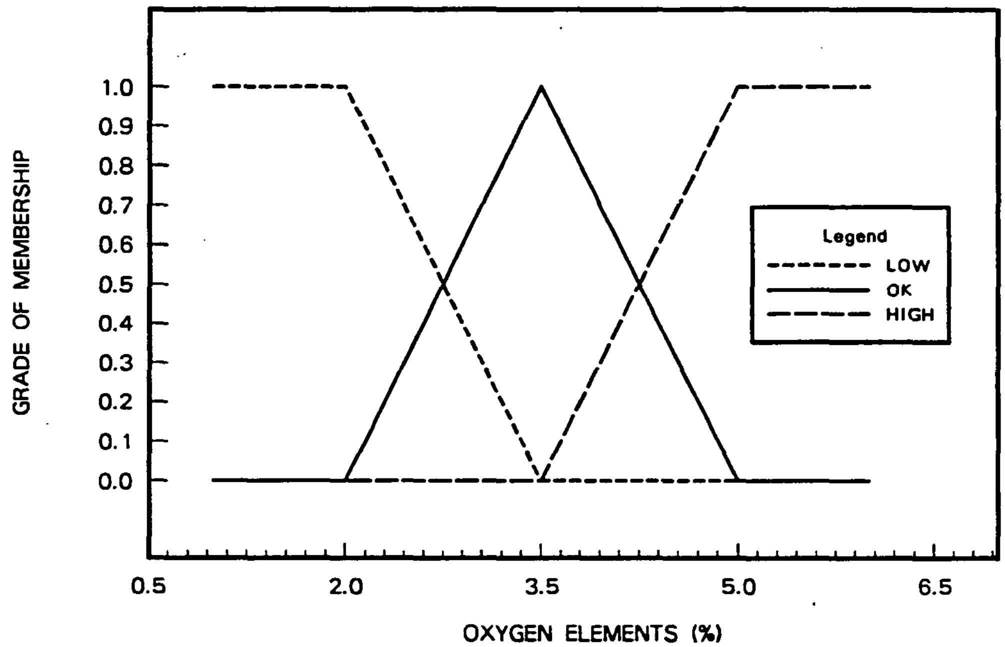

Figure 11.5: Membership Functions for "Low", "Ok", and "High" Oxygen

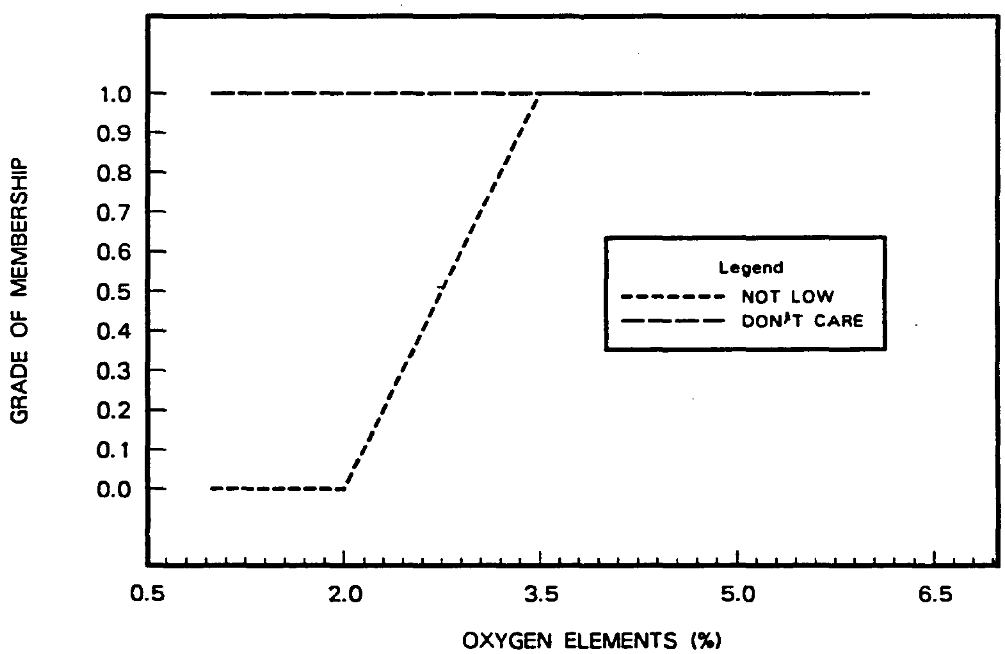

Figure 11.6: Membership Functions for "Not Low" and Don't Care Conditions for Oxygen

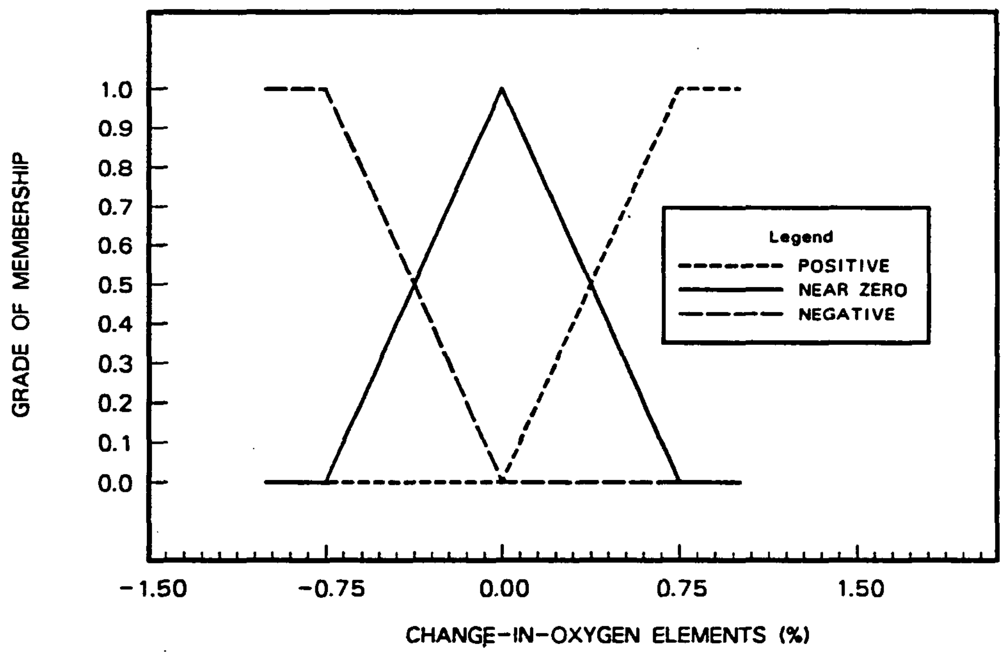

Figure 11.7: Membership Functions for "Positive", "Near Zero", and "Negative" Δ Oxygen

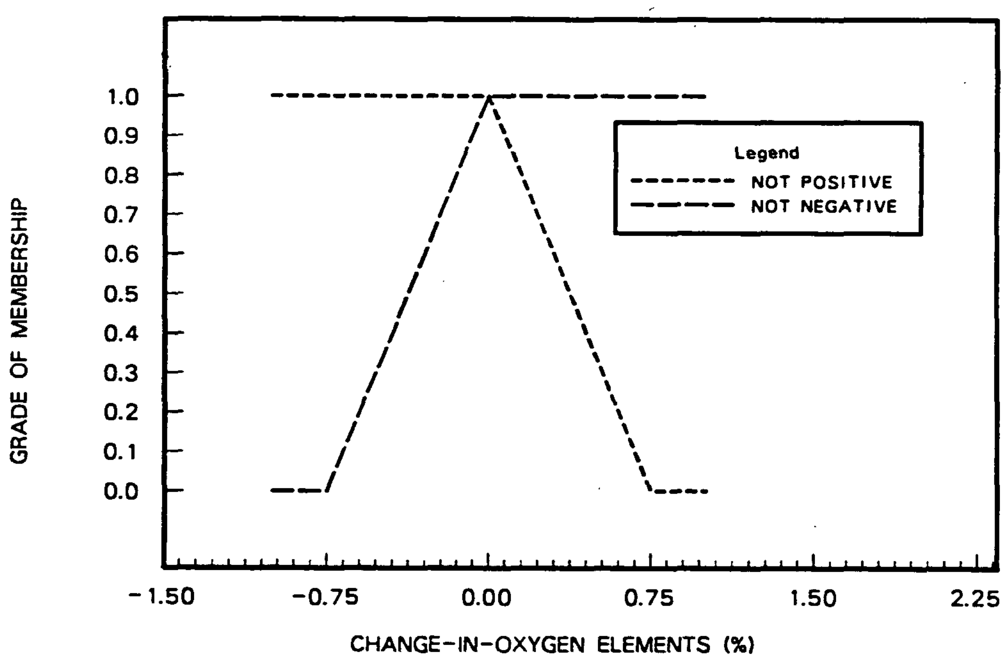

Figure 11.8: Membership Functions for "Not Positive" and "Not Negative" Δ Oxygen

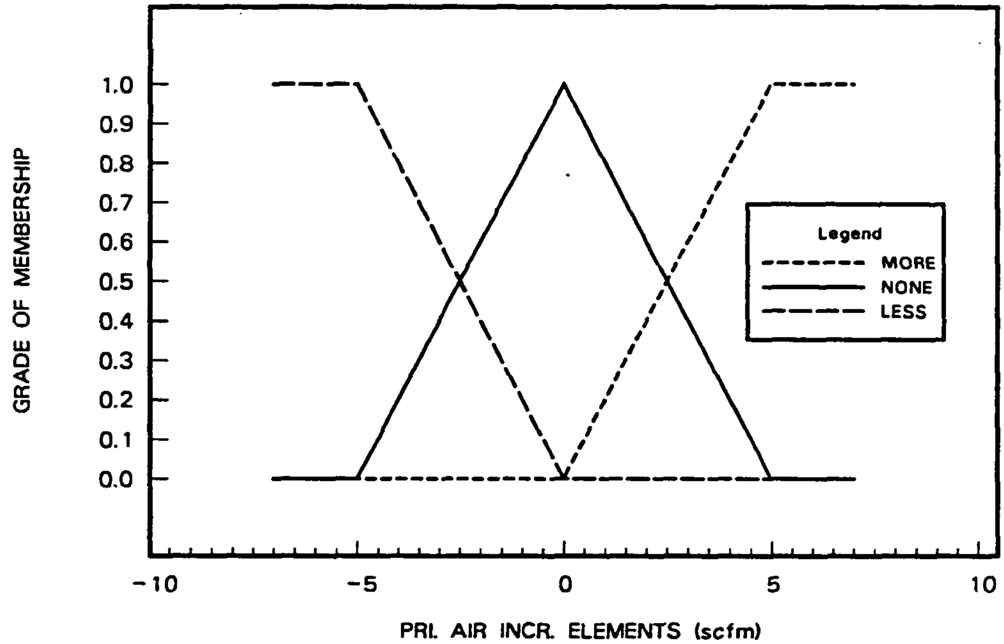

Figure 11.9: Membership Functions for "More", "None", and "Less" Primary Air Flow Increment

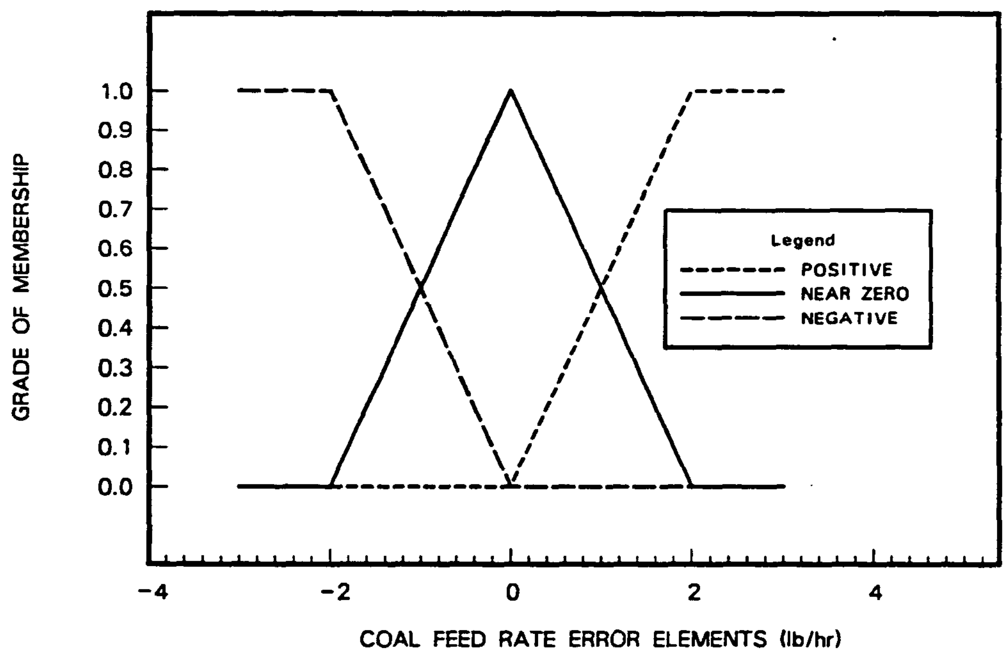

Figure 11.10: Membership Functions for "Positive", "Near Zero", and "Negative" Coal Feed Rate Error

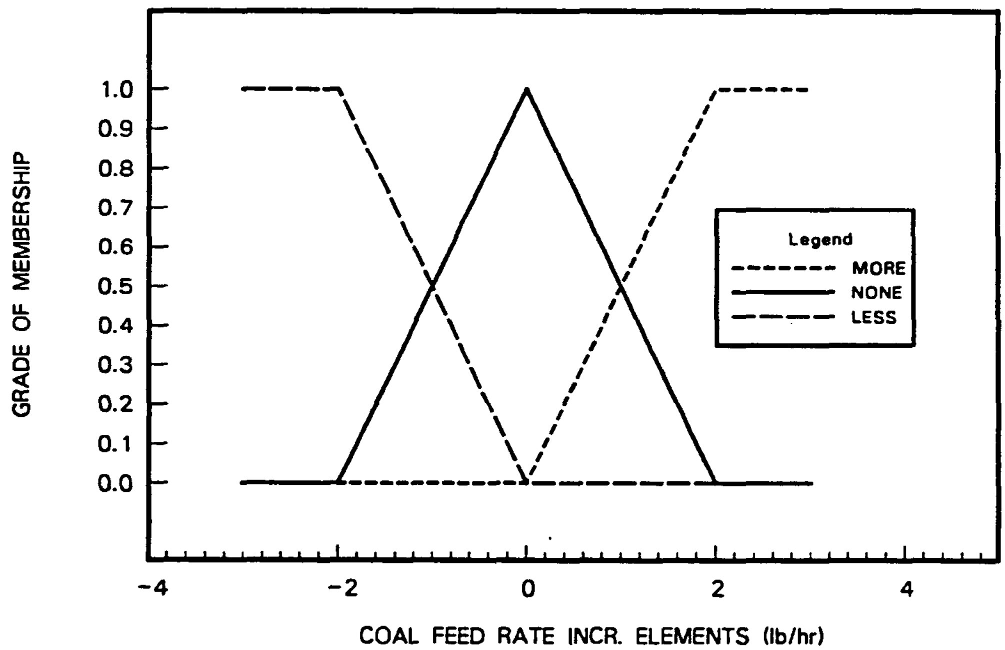

Figure 11.11: Membership Functions for "More", "None", and "Less" Coal Feed Rate Increment

12. APPENDIX E. PI CONTROLLER DESCRIPTION

PI controller gains (see Table 12.1) are designed by the Ziegler-Nichols transient-response method. Gain scheduling is employed for oxygen control as constant gains will not satisfactorily control oxygen for both operating régimes of LP gas preheating and coal feeding. Scheduling is accomplished by linearly interpolating between the two sets of gains over a two-minute time interval after coal feeding is initiated. Interpolation provides a smooth transition so as not to "shock" the system into unstable behavior. Control actions are enacted as fast as possible (approximately every 0.6 seconds) since gains from the Ziegler-Nichols method are determined for continuously-operated systems[1], and the integral of error term is determined by trapezoidal approximation.

Table 12.1: PI Controller Gains for Different Operating Régimes

Controller Gains For:		*Operating Régimes* LP Gas Preheating	Coal Feeding
PRIMARY AIR	Proportional	1.25	2.87
	Integral	0.0250	0.0840
SECONDARY AIR	Proportional	-0.100	
	Integral	-0.000833	

[1]Ziegler-Nichols gains can be used with success on sampled-data systems with moderate-to-fast sampling rates.